Retrieval-Constrained Policy Optimization for Attack Technique Extraction from Cyber Threat Intelligence

Jiayun Zhang
jiayunz@amazon.com
Amazon Web Services
New York, New York, USA

Junshen Xu
jsxu@amazon.com
Amazon Web Services
Boston, Massachusetts, USA

Zejun Xie*
zx180@cs.rutgers.edu
Rutgers University
Piscataway, New Jersey, USA

Yi Fan
fnyi@amazon.com
Amazon Web Services
New York, New York, USA

Abstract

Mapping cyber threat intelligence (CTI) text to MITRE ATT&CK techniques is essential for structured threat analysis, yet manual annotation is costly and does not scale. The ATT&CK taxonomy comprises several hundred attack techniques, and a single CTI passage may describe multiple techniques, making accurate and complete extraction challenging. Existing automated approaches fall short in different ways: multi-label classifiers struggle with severe class imbalance and the large label space, while LLM-based methods—retrieval pipelines and fine-tuned generators—optimize token-level objectives that treat technique annotation as sequence generation rather than set prediction, lacking direct supervision on whether the predicted technique set is correct and complete. We propose TTP-R1, a two-stage framework that combines retrieval-augmented supervised fine-tuning (SFT) with reinforcement learning using verifiable rewards (RLVR). A hybrid retriever first narrows the large label space to a candidate set, and a fine-tuned LLM learns to select the correct techniques. We then apply Group Relative Policy Optimization with a decomposed reward that directly supervises the precision, recall, and output format of the predicted technique set. Across four CTI benchmarks, TTP-R1 achieves the best average F1, improving sub-technique-level F1 by 7.4 percentage points over Claude Sonnet 4.5 with retrieval augmentation, while running 28× faster when served as an 8B-parameter model on a single GPU.

CCS Concepts

• **Computing methodologies** → **Information extraction**; **Reinforcement learning**; • **Information systems** → *Retrieval models and ranking*; • **Security and privacy**;

Keywords

Cyber Threat Intelligence, Attack Technique Extraction, Retrieval-Augmented Generation, Reinforcement Learning

1 Introduction

Cyber threat intelligence (CTI) reports describe adversarial behaviors in natural language—the tools, methods, and objectives employed by attackers. While rich in detail, these unstructured narratives are difficult to operationalize at scale. To convert them into structured, machine-readable representations, security analysts map CTI text to the MITRE ATT&CK framework [26], a standardized taxonomy that organizes adversary behaviors into *tactics* (high-level goals such as Lateral Movement), *techniques* (specific methods to achieve those goals), and *procedures* (real-world implementations of techniques). The taxonomy defines over five hundred techniques and sub-techniques, organized hierarchically. The resulting structured annotations enable downstream applications such as threat hunting [28], detection rule generation [24], and intelligence sharing [14], and are widely used in production security systems. Figure 1 gives an example of technique mapping from CTI and downstream applications. As the volume of CTI publications grows, manually annotating techniques becomes a bottleneck because it requires both breadth of knowledge and fine-grained reasoning, and cannot scale to match the pace of new threat reports, motivating the need for automated extraction methods.

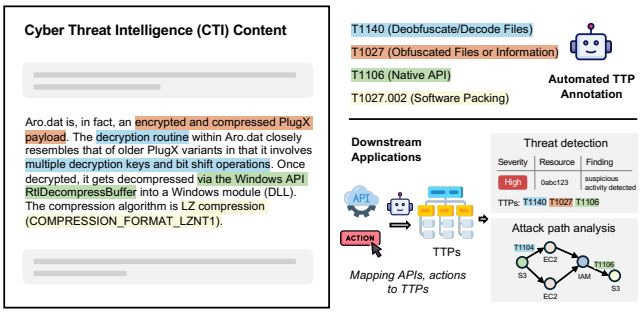

Figure 1: Example of attack technique extraction from Cyber Threat Intelligence (CTI). A CTI passage describing adversary behavior is mapped to MITRE ATT&CK techniques, including parent techniques (e.g., T1140) and sub-techniques (e.g., T1027.002). The extracted techniques feed downstream applications such as threat detection and attack path analysis.

Existing approaches to automated attack technique extraction fall into two broad categories. Classification-based methods train multi-class or multi-label classifiers to predict technique identifiers directly. They struggle with severe class imbalance and data scarcity given the large label space [7, 16, 20, 31]. Retrieval-based methods rank candidate techniques by similarity to the input text [3, 13], and recent hybrid pipelines combine retrieval with Large Language

*Work done during internship at Amazon Web Services.

Model (LLM)-based generation to achieve strong results on single-label benchmarks [15, 29]. Despite this progress, a critical gap remains in the multi-label setting, where real-world CTI sentences often describe multiple techniques simultaneously. Prompting frontier LLMs, which have broad knowledge of attack patterns, leads to over-prediction: the model infers techniques that commonly co-occur with those described in the text, even when they are not explicitly supported by the input. Fine-tuning LLMs with token-level cross-entropy improves precision but produces conservative, single-label outputs that miss co-occurring techniques—a consequence of treating technique extraction as sequence generation rather than set prediction. In neither case is there direct supervision on whether the predicted technique set is correct and complete.

We propose TTP-R1, a two-stage framework that addresses this gap by combining retrieval-augmented supervised fine-tuning (SFT) with reinforcement learning using verifiable rewards (RLVR). In the first stage, a hybrid retriever narrows the large label space to a manageable candidate set, and an LLM is fine-tuned to select techniques from this set. In the second stage, we apply Group Relative Policy Optimization (GRPO) [25] that provides separate signals for recall, precision, and output format of the predicted technique set, giving the optimizer direct supervision that token-level objectives lack. To handle the challenges of multi-reward optimization, we decouple the normalization of each reward channel [17], ensuring that each reward component contributes proportionally to the policy gradient. A selective training strategy focuses RL updates on examples that remain difficult after SFT, preventing easy instances from contributing uninformative gradients.

Our contributions are as follows:

- We formulate attack technique extraction as a *retrieval-then-select* task and show that augmenting SFT with reinforcement learning can directly supervise set-level metrics, addressing the multi-label reasoning gap left by token-level training objectives.
- We design a decomposed verifiable reward with separate precision, recall, and format channels, combined via decoupled normalization, enabling fine-grained control over multi-label prediction quality.
- Across four CTI benchmarks, TTP-R1 achieves the best average F1. At the sub-technique level, it outperforms Claude Sonnet 4.5 with the same RAG prompt by 7.4 percentage points, while reducing latency by 28× using an 8B model on a single GPU. Ablation studies validate the contribution of each component and demonstrate generalization across two base model families.

2 Related Work

2.1 TTP Extraction from Cyber Threat Intelligence

Mapping CTI text to MITRE ATT&CK tactics, techniques, and procedures (TTPs) has been approached from several angles [5]. Early rule-based systems such as TTPDrill [11] relied on hand-crafted patterns and NLP pipelines, which were precise but brittle and difficult to maintain as the ATT&CK taxonomy evolved. Subsequent work framed the problem as multi-class classification. You et al. [31] trained classifiers to predict technique IDs directly, but these methods struggled with severe class imbalance given the hundreds of possible labels. Knowledge-enhanced approaches sought to mitigate this: AttacKG [16] incorporated knowledge graphs to capture inter-technique relationships, while Nguyen et al. [20] applied noise-contrastive estimation to learn more discriminative technique representations. Retrieval-based methods offer an alternative by ranking candidate techniques against the input text. LADDER [3] used semantic similarity for candidate ranking, and Text2TTP [13] introduced hierarchical re-ranking with dual-encoder learning. More recently, hybrid retrieve-then-generate pipelines have emerged. IntelEX [29] combined retrieval with GPT-style zero-shot reasoning, while TechniqueRAG [15] anchored a fine-tuned LLM with retrieved evidence and achieved strong results on single-label benchmarks. However, these methods optimize token-level objectives and do not explicitly address the multi-label setting where multiple techniques co-occur in a single text—a scenario that remains challenging in practice. This gap is visible in prior results, where TechniqueRAG [15] performs strongly on single-label datasets but degrades substantially on the multi-label scenario. Our work builds on the retrieve-then-select paradigm of TechniqueRAG but introduces an RL stage, specifically targeting the multi-label reasoning gap left by prior approaches.

2.2 LLMs for Information Extraction in Security

LLMs have shown promise across a range of security-related NLP tasks, including threat entity extraction, vulnerability analysis, and malware classification [8]. LLMCloudHunter [24] demonstrated that LLMs can automatically extract API calls and indicators of compromise from CTI reports to generate detection rules for cloud environments. For technique identification specifically, zero-shot prompting of frontier models such as GPT-4 has been explored [29], but performance remains limited due to the domain-specific nature of the ATT&CK taxonomy and the tendency of general-purpose LLMs to hallucinate plausible but incorrect technique IDs. Fine-tuning on domain data substantially improves accuracy: TechniqueRAG [15] demonstrated that even a modest number of labelled examples suffices to adapt an instruction-tuned LLM for this task. A common limitation of these supervised approaches is that the cross-entropy training objective treats each token independently and does not directly optimize the set-level metrics (precision, recall) that practitioners care about. Our method addresses this by adding an RL fine-tuning stage on top of supervised fine-tuning.

2.3 Reinforcement Learning for LLM Reasoning

Reinforcement learning has become a key technique for aligning LLMs beyond what supervised fine-tuning alone can achieve. RLHF [21] uses human preference judgements to train a reward model, but collecting such annotations is expensive and difficult to scale. Reinforcement learning with verifiable rewards (RLVR) replaces the learned reward model with a programmatic verifier that scores outputs against ground truth. DeepSeek-R1 [9] and DeepSeek-Math [25] demonstrated that RLVR with Group Relative Policy Optimization (GRPO) can substantially improve LLM reasoning on mathematical and code tasks. GRPO eliminates the need for a separate value network by computing advantages from multiple sampled completions per prompt, making it both simpler and more stable than standard PPO for language generation. Rank-R1 [33]

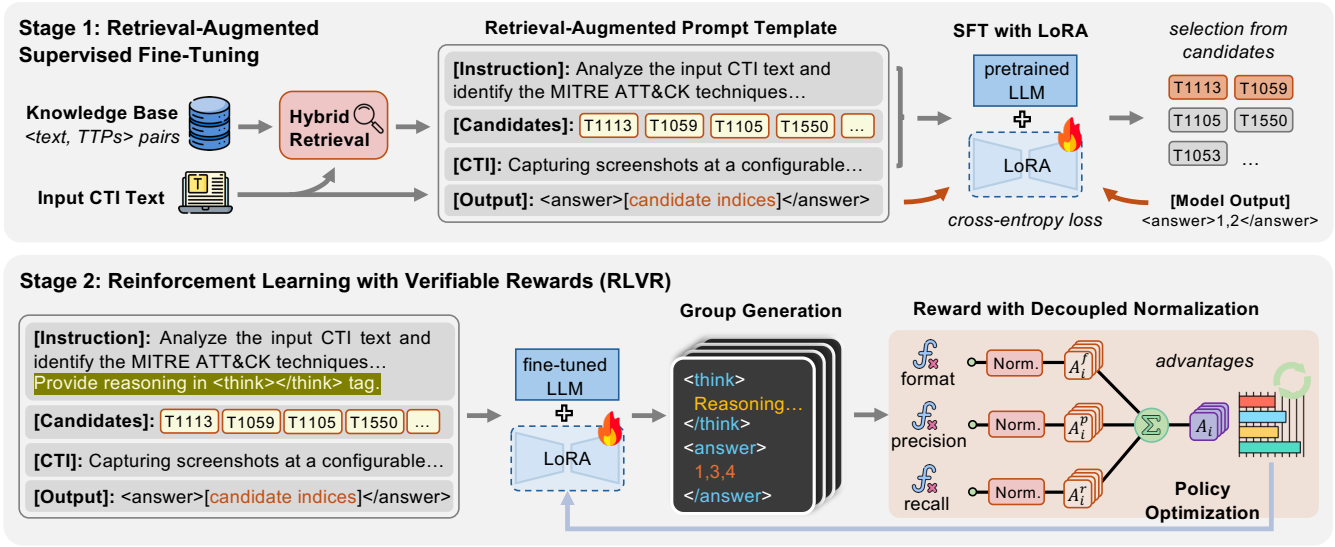

Figure 2: Overview of the TTP-R1 framework. Stage 1 uses a hybrid BM25 + embedding retriever to narrow the 500+ ATT&CK taxonomy to a candidate set, then fine-tunes an LLM to select correct techniques from the candidates. Stage 2 applies GRPO with decomposed verifiable rewards that separately supervise precision, recall, and output format. Applied selectively on examples that remain difficult after SFT.

extended this paradigm to document re-ranking, showing that binary relevance rewards suffice to match fully supervised rerankers. GDPO [17] further improved multi-reward GRPO training by normalizing each reward channel independently before aggregation, preventing high-variance components from dominating the policy gradient. In cybersecurity, a recent work Minerva [2] applies RLVR to a broad suite of CTI subtasks, including technique extraction, vulnerability mapping, and threat actor attribution. Its technique extraction predicts a single technique ID per input, rewarded by match of the technique ID, with partial credit at the base-technique level. Our work applies RLVR with GRPO to multi-label technique extraction, where multiple techniques co-occur in a single text, and designs a decomposed reward with separate precision and recall channels combined via decoupled normalization, providing the optimizer with independent signals for correctness and completeness.

3 Methodology

We present TTP-R1 in two stages. Figure 2 gives an overview. In Stage 1, we formulate attack technique extraction as a *retrieval-then-select* task: a hybrid retriever narrows the large label space to a manageable candidate set, and a supervised fine-tuned (SFT) LLM learns to select the correct techniques from that set. In Stage 2, we apply reinforcement learning with decomposed, verifiable rewards to directly supervise set-level correctness and completeness on examples that remain difficult after SFT.

3.1 Stage 1: Retrieval-Augmented Supervised Fine-Tuning

Problem formulation. Given a CTI text x, the goal is to predict the set of MITRE ATT&CK technique identifiers $Y \subseteq \mathcal{Y}$ described in x, where $\mathcal{Y}$ is the full taxonomy of techniques and sub-techniques. Because $|\mathcal{Y}|$ is large (several hundred entries), directly generating technique IDs in an open-ended fashion is prone to hallucination. We therefore decompose the problem into two steps: (i) retrieve a small candidate set $\mathcal{T}_x \subset \mathcal{Y}$ likely to contain the correct labels, and (ii) select the correct subset $\hat{Y} \subseteq \mathcal{T}_x$ with an LLM.

Hybrid candidate retrieval. We maintain a corpus C of labelled CTI sentences from the training data, each associated with one or more technique identifiers. Given input x, we retrieve relevant corpus entries using a hybrid of BM25 [23] and ATT&CK-BERT [1], a Sentence-BERT [22] encoder pretrained with a Siamese objective on attack-action text so that sentences describing semantically similar adversary behaviors map to nearby embeddings. Both retrievers return ranked lists over C and we merge them by taking the minimum:

$$\text{rank}(c) = \min\big(\text{rank}_{\text{BM25}}(c),\ \text{rank}_{\text{BERT}}(c)\big), \tag{1}$$

and sorting accordingly. After removing entries whose label sets are identical (deduplication), we retain the top-k unique examples. The candidate set $\mathcal{T}_x = \{t_1, \ldots, t_m\}$ is formed by collecting all technique labels from these k examples in retrieval-rank order, preserving first-seen ordering and capping at m labels. The hybrid design combines the lexical precision of BM25 with the semantic generalisation of dense retrieval, improving candidate recall over either method alone.

Prompt construction and supervised fine-tuning. Each training example (x, Y) is converted into an instruction-following sample. The prompt comprises three parts: (i) a system-level role description instructing the model to act as a MITRE ATT&CK analyst, (ii) the candidate list $\mathcal{T}_x$ with technique IDs, names, and brief descriptions, and (iii) the CTI text x. The target output enumerates the

SFT Prompt Template

[User]
You are a cybersecurity expert specializing in the MITRE ATT&CK framework. Your task is to analyze cyber threat descriptions and identify the associated MITRE ATT&CK techniques. The MITRE ATT&CK framework is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations. This framework is used for developing specific threat models and methodologies in the private sector, government, and cybersecurity product and service community. Analyze the input cyber threat description and list the relevant MITRE ATT&CK techniques. Select the MITRE ATT&CK techniques from the following list and output their number in <answer> </answer> tags.

A list of techniques that may be relevant:
[1] T1547.001 Boot or Logon Autostart Execution: [description]
[2] T1059.001 Command and Scripting Interpreter: [description]
[...m candidates total...]

Input cyber threat description:
[CTI text x]

[Assistant]
<answer>
1
</answer>

Figure 3: SFT prompt template used in Stage 1 training. The model receives a cybersecurity expert role description, a CTI text passage, and a numbered candidate list populated by the hybrid retriever. The target output selects the correct candidates by their indices within <answer> tags.

RL Prompt Template (additions over SFT prompt highlighted in blue)

[User]
You are a cybersecurity expert specializing in the MITRE ATT&CK framework. Your task is to analyze cyber threat descriptions and identify the associated MITRE ATT&CK techniques. *[...framework description...]* Analyze the input cyber threat description and list the relevant MITRE ATT&CK techniques. Provide your reasoning process within <think> </think> tags. Select the MITRE ATT&CK techniques from the following list and output their number in <answer> </answer> tags.

A list of techniques that may be relevant:
[1] T1547.001 Boot or Logon Autostart Execution: [description]
[2] T1059.001 Command and Scripting Interpreter: [description]
[...m candidates total...]

Input cyber threat description:
[CTI text x]

Figure 4: RL prompt template used in Stage 2 training. Compared to the SFT prompt (Figure 3), the model is additionally instructed to produce a reasoning trace within <think> tags before committing to its answer.

indices of the correct candidates $Y \cap \mathcal{T}_x$ within structured <answer> tags. Training examples for which the retriever fails to surface any ground-truth label ($Y \cap \mathcal{T}_x = \emptyset$) are discarded, as they provide no useful supervision signal. We fine-tune a pre-trained instruction-tuned LLM on these samples with Low-Rank Adaptation (LoRA) [10] by minimizing the standard causal language modelling loss:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,\,Y)}\left[\sum_{t=1}^{|y^*|} \log \pi_\theta\big(y_t^* \mid y_{<t}^*,\, x,\, \mathcal{T}_x\big)\right], \tag{2}$$

where $y^* =$ <answer> $\{i : t_i \in Y \cap \mathcal{T}_x\}$ </answer> is the target sequence. This produces the SFT policy π_{SFT}. Figure 3 shows the complete prompt template.

3.2 Stage 2: Reinforcement Learning with Verifiable Rewards

The SFT objective optimizes token-level cross-entropy, which does not directly correspond to set-level evaluation metrics such as precision and recall. It penalises the model for generating correct techniques in the wrong order and offers no direct mechanism to supervise whether the predicted set is correct and complete; in practice, this causes the SFT model to favour conservative, single-label predictions that achieve high precision but miss ground-truth techniques. To bridge this gap, we further train π_{SFT} with reinforcement learning using rewards that are automatically computed from the ground-truth labels.

Selective training. We first run inference with π_{SFT} on the training set multiple times and measure per-example F1. Only examples whose average F1 falls below a threshold τ are retained for RL training. This focuses optimization on the cases where the SFT model still produces errors, avoiding redundant gradient updates on already-solved instances and improving sample efficiency.

Structured generation. During RL, the model is prompted to first produce a free-form reasoning trace enclosed in <think> … </think> tags, followed by its final predictions in an <answer> … </answer> block. Only the content of the <answer> block is used for reward computation; the reasoning trace serves as a chain-of-thought scaffold [27] that empirically improves answer quality without being directly supervised. RL prompt template is shown in Figure 4.

Reward design. Given a completion, we extract the predicted set $\hat{Y}$ from the answer block and compute three reward signals against the ground truth Y:

- **Precision reward r_P.** We evaluate precision at both the technique level and the sub-technique level, and average the two.

$$r_P = \frac{1}{2}\left(\frac{|\hat{Y}_T \cap Y_T|}{|\hat{Y}_T|} + \frac{|\hat{Y}_S \cap Y_S|}{|\hat{Y}_S|}\right), \tag{3}$$

  where subscripts T and S denote parent-technique and sub-technique granularities, respectively.

- **Recall reward r_R.** Similarly, we evaluate recall at both technique and the sub-technique level, and take the average. Defined analogously over ground-truth set sizes:

$$r_R = \frac{1}{2}\left(\frac{|\hat{Y}_T \cap Y_T|}{|Y_T|} + \frac{|\hat{Y}_S \cap Y_S|}{|Y_S|}\right). \tag{4}$$

- **Format reward r_F.** A discrete score that verifies whether the output contains well-formed `<think>` and `<answer>` blocks with non-empty content, encouraging adherence to the structured output format.

The final reward is a weighted sum:

$$r = w_P\, r_P + w_R\, r_R + w_F\, r_F. \tag{5}$$

A key design choice is to *decompose* precision and recall into separate reward channels rather than collapsing them into a single F1 score. A single F1 reward provides only one scalar signal per completion, making it difficult for the optimizer to distinguish whether a low score stems from over-prediction (low precision) or under-prediction (low recall). By decomposing, the optimizer receives independent gradient signals for each axis, enabling finer control over set-level metrics through the weights w_P and w_R. Moreover, evaluating at both granularities simultaneously encourages the model to predict the correct parent technique even when it is uncertain about the specific sub-technique.

Policy optimization with reward-decoupled normalization. We optimize the policy π_θ (initialized from π_{SFT}) using Group Relative Policy Optimization (GRPO) [25]. For each prompt x, it samples a group of G completions from π_θ. In standard GRPO, the advantage for the i-th completion is $A_i = (r_i - \mu)/\sigma$, where μ and σ are the mean and standard deviation of the scalar rewards within the group. When multiple reward channels are combined into a single scalar before normalization, distinct reward combinations can collapse into identical advantage values, losing the training signal that distinguishes them. Inspired by Liu et al. [17], we address this by decoupling the normalization across reward channels: each channel is normalized independently within the group before weighted aggregation. Let $r_i^{(j)}$ denote the j-th reward channel for completion i, and let $\mu^{(j)}, \sigma^{(j)}$ be the within-group mean and standard deviation of that channel. The reward-decoupled advantage is:

$$\hat{A}_i = \text{BN}\left(\sum_j w_j \cdot \frac{r_i^{(j)} - \mu^{(j)}}{\sigma^{(j)} + \delta}\right), \tag{6}$$

where $\text{BN}(\cdot)$ denotes batch-level mean-variance normalization and δ is a small constant for numerical stability. This preserves the relative differences across reward channels regardless of their raw scale or variance.

The full training objective is:

$$\mathcal{L}(\theta) = -\mathbb{E}_x\left[\frac{1}{G}\sum_{i=1}^{G}\min\Big(\rho_i\,\hat{A}_i,\ \text{clip}(\rho_i, 1-\epsilon, 1+\epsilon)\,\hat{A}_i\Big) - \beta\, D_{\text{KL}}(\pi_\theta \| \pi_{\text{SFT}})\right], \tag{7}$$

where $\rho_i = \pi_\theta(y_i \mid x)/\pi_{\text{old}}(y_i \mid x)$ is the importance-sampling ratio, ϵ controls ratio clipping, and β weights a KL penalty that prevents the policy from drifting too far from the SFT initialization.

Table 1: Statistics for the four CTI benchmarks.

Dataset	# Samples		Avg. Labels		Avg. Words		Unique TTPs	
	Train	Test	Train	Test	Train	Test	Train	Test
TRAM	4,072	725	1.16	1.17	21.0	21.3	184	127
Procedures	9,956	1,767	1.01	1.00	13.4	13.4	474	300
Derived Procedures	2,998	521	1.23	1.20	67.3	64.9	359	194
Expert	540	157	1.42	3.32	38.2	71.9	251	152

After RL training, the LoRA adapter is merged into the base model to produce the final policy $\pi_{\text{TTP-R1}}$.

4 Experiments

4.1 Datasets

We evaluate on four benchmarks[1] for MITRE ATT&CK technique annotation from CTI text, adopting the same splits as Lekssays et al. [15]. We use the combined train and dev splits for fine-tuning. Table 1 summarizes the statistics. The descriptions are as follows:

- **TRAM** [6] is a large publicly available manually curated dataset, released by the Center for Threat-Informed Defense. It comprises short text fragments extracted from threat reports, each labeled with ATT&CK (sub-)technique IDs. As noted by Nguyen et al. [20], TRAM covers only about one-third of the ATT&CK taxonomy and contains relatively noisy labels. The fragments often lack surrounding context (e.g., "can download and execute a file from given URL" with no subject indicating what tool or malware performs the action), making it difficult for technique assignment.
- **Procedures** [20] consists of one-sentence expert-written summaries of how a technique is implemented in real-world attacks, collected from the ATT&CK knowledge base (v12.0) procedure examples. This is effectively a single-label dataset. With 300 unique sub-techniques in the test split, it tests breadth of coverage but is relatively straightforward since sentences closely mirror technique definitions.
- **Derived Procedures** [20] complements the Procedures dataset with longer passages that align to threat report writing style. These are evidential paragraphs sourced from the URL references cited in ATT&CK procedure examples, identified via per-document search. Unlike Procedures, these passages require reasoning over implicit adversary behaviors rather than matching explicit technique descriptions.
- **Expert** [20] is a paragraph-level dataset annotated by five CTI experts, designed to closely emulate real-world technique extraction scenarios. Unlike the sentence-focused datasets above, Expert covers entire paragraphs, and annotations are inherently multi-label (avg. 3.32 labels, up to 18 in the test split). This is the most challenging benchmark, testing the model's ability to identify multiple co-occurring techniques.

4.2 Compared Methods

We compare TTP-R1 against three categories of baselines:

[1]Datasets are available at: https://github.com/tumeteor/mitre-ttp-mapping

Table 2: Technique-level evaluation (%). Sub-technique IDs are truncated to their parent before computing metrics. Best results in bold, second-best underlined. Avg. macro-F1 and Rank are computed across all four datasets. TTP-R1 achieves the highest average F1 and the best average rank, outperforming frontier LLMs with RAG and domain-specific baselines.

Model	TRAM			Procedures			Derived Procedures			Expert			Avg.	
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	F1	Rank
DeepSeek-R1	49.2	72.8	58.7	64.4	85.9	73.6	28.0	66.8	39.4	43.6	59.1	50.2	55.5	7.5
w/ RAG	54.7	80.0	65.0	68.5	91.9	78.5	31.1	71.6	43.3	45.9	59.3	51.8	59.7	4.5
Claude Sonnet 4.5	52.9	76.3	62.5	67.6	84.8	75.2	28.2	72.6	40.6	46.5	67.5	55.1	58.3	5.5
w/ RAG	59.4	78.8	67.8	73.7	**92.9**	82.2	31.8	73.6	44.4	59.0	70.3	**64.2**	64.7	3.2
Qwen3-8B	20.4	28.6	23.9	25.0	32.2	28.1	12.6	29.0	17.5	20.2	25.6	22.6	23.0	12.5
w/ RAG	52.3	73.3	61.0	67.0	85.9	75.3	32.4	58.5	41.7	47.1	50.1	48.6	56.6	6.5
Ministral-8B	13.5	26.1	17.8	16.0	28.6	20.5	8.5	21.9	12.2	11.0	15.5	12.9	15.8	14.0
w/ RAG	46.8	78.2	58.6	53.3	79.7	63.9	26.3	71.0	38.4	31.0	50.5	38.4	49.8	10.2
Nova 2 Lite	20.6	48.2	28.9	26.7	54.9	35.9	8.2	34.8	13.3	13.8	33.4	19.6	24.4	12.5
w/ RAG	51.6	71.5	59.9	62.0	79.8	69.8	28.9	61.0	39.3	45.6	48.3	46.9	54.0	8.5
IntelEX	35.9	**85.4**	50.6	47.0	91.4	62.1	16.0	**82.4**	26.7	32.5	**82.9**	46.7	46.5	10.5
TechniqueRAG	76.0	72.1	74.0	**91.1**	91.1	**91.1**	46.8	42.9	44.8	**75.2**	37.7	50.2	65.0	3.2
TTP-R1 w/o RL	**82.2**	79.2	80.6	89.9	89.8	89.9	**52.7**	49.0	50.8	67.8	33.6	45.0	66.6	4.2
TTP-R1	80.2	84.5	**82.3**	89.5	90.5	90.0	48.1	57.8	**52.5**	64.8	41.5	50.6	**68.9**	**2.0**

Table 3: Sub-technique-level evaluation (%). Predictions must match the full sub-technique ID. TTP-R1's advantage grows at this finer granularity (+7.4 F1 over Claude Sonnet 4.5 w/ RAG, vs. +4.2 at technique-level).

Model	TRAM			Procedures			Derived Procedures			Expert			Avg.	
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	F1	Rank
DeepSeek-R1	36.2	57.8	44.6	55.7	76.3	64.4	21.6	54.9	31.0	34.4	44.6	38.8	44.7	9.2
w/ RAG	41.2	68.7	51.5	61.1	87.7	72.0	25.7	68.1	37.3	41.2	50.3	45.3	51.5	4.5
Claude Sonnet 4.5	43.2	67.5	52.7	59.8	79.9	68.4	22.2	64.4	33.0	43.0	57.7	49.3	50.9	5.5
w/ RAG	48.8	76.0	59.4	61.4	**88.9**	72.6	25.1	69.2	36.8	53.8	61.0	**57.2**	56.5	3.5
Qwen3-8B	14.7	23.4	18.1	12.3	16.9	14.2	5.6	14.9	8.2	15.6	18.4	16.9	14.3	12.5
w/ RAG	38.9	61.8	47.7	60.2	81.2	69.1	27.0	53.2	35.8	41.6	42.3	42.0	48.6	6.8
Ministral-8B	8.6	20.8	12.2	6.4	14.7	8.9	3.1	11.3	4.8	8.3	9.8	9.0	8.7	14.0
w/ RAG	36.4	69.0	47.7	44.0	71.7	54.5	21.2	64.5	31.9	26.8	41.6	32.6	41.7	9.8
Nova 2 Lite	15.9	40.8	22.9	16.1	38.0	22.6	3.8	20.5	6.4	10.5	24.4	14.7	16.6	12.5
w/ RAG	41.6	65.0	50.7	53.3	76.0	62.7	24.0	56.3	33.6	40.6	41.4	41.0	47.0	7.8
IntelEX	28.3	**83.3**	42.2	35.5	87.0	50.4	12.2	**79.8**	21.2	29.8	**79.7**	43.4	39.3	9.5
TechniqueRAG	72.7	68.7	70.7	**91.1**	88.1	**88.1**	40.1	35.9	37.9	**70.1**	30.2	42.2	59.7	3.2
TTP-R1 w/o RL	**78.8**	75.9	77.3	85.7	85.7	85.7	**46.9**	44.0	45.4	65.0	27.3	38.5	61.7	4.2
TTP-R1	76.4	81.3	**78.7**	85.1	86.5	85.8	42.0	52.6	**46.7**	60.8	34.7	44.2	**63.9**	**2.0**

(1) **Frontier LLMs.** We evaluate DeepSeek-R1 [9] and Claude Sonnet 4.5 [4], both in zero-shot and RAG-augmented settings. These represent the strong general-purpose reasoning models available at the time of evaluation. In the zero-shot setting, models receive only the task instruction and input text. In the RAG setting, we provide the same retrieved candidate list used by TTP-R1—up to 25 techniques with their IDs and short descriptions—in the prompt, asking the model to select from among them.

(2) **Open-weight LLMs.** We evaluate Qwen3-8B [30], Ministral-8B-Instruct [18], and Amazon Nova 2 Lite [12], each in zero-shot and RAG-augmented settings. These models match TTP-R1 in parameter count (8B) and serve as direct comparisons to isolate the effect of domain-specific fine-tuning from model scale. Ministral-8B is the same base model used by TTP-R1 and TechniqueRAG, enabling controlled comparison.

(3) **Domain-Specific methods.** IntelEX [29] is a training-free pipeline that combines retrieval-augmented candidate generation with LLM-as-a-judge validation. For each candidate technique retrieved from the MITRE knowledge base, an LLM independently determines whether it is present in the input text

by generating a YES/NO judgment with justification. We re-implement IntelEX using Claude Sonnet 4.5 as the backbone LLM (the original uses GPT-4o-mini). The pipeline validates all RAG candidates individually, which yields high recall but low precision due to the inability to jointly reason about which subset of candidates is most relevant. TechniqueRAG [15] is the prior state-of-the-art framework for this task. It retrieves candidate text–technique pairs via BM25, re-ranks them using DeepSeek V3 with a domain-specific prompting framework that decomposes queries into attack steps and evaluates sub-technique relevance, then fine-tunes Ministral-8B as the generator on the top re-ranked few-shot examples. We report their published results on the same test splits. Since TechniqueRAG does not have results on Derived Procedures, we run their released model on this dataset using the same evaluation protocol for a complete comparison.

For all models in (1) and (2), we show the performance both without and with RAG. In the RAG setting, we provide the same retrieved candidate list (up to 25 techniques) used by TTP-R1.

4.3 Implementation Details

Retrieval. We use BM25 (Okapi, k_1=1.6, b=0.75) combined with ATT&CK-BERT [1], a domain-pretrained Sentence-BERT encoder, and FAISS for dense retrieval. We retrieve k=25 candidate techniques per query, presenting each as a numbered option with its technique ID and description. The retrieval recall at the sub-technique level is 97.3% on TRAM, 96.9% on Procedures, 89.7% on Derived Procedures, and 78.6% on Expert. As the LLM selects only from the retrieved candidate list, this caps the model's attainable recall. Training examples for which no correct technique appears in the candidate list provide no useful supervision and are discarded before SFT (786 out of 17,566), resulting in 16,780 training samples.
Training. We fine-tune Ministral-8B-Instruct [18] in two stages.

- **SFT**: LoRA (r=32, α=64) on 16,780 training samples for 3 epochs with learning rate 10^{-4} and effective batch size 64.
- **RL**: Initialized from the SFT checkpoint, we apply GRPO [25] with LoRA (r=8, α=16) for 1 epoch. We sample G=16 completions per prompt at temperature 1.4 and optimize a reward combining precision and recall: $r = 0.45P + 0.45R + 0.1r_F$. A selective training threshold τ=0.99 skips prompts where all completions already achieve near-perfect F_1.

All training uses DeepSpeed ZeRO-2 on a `g6e.48xlarge` instance with 8× NVIDIA L40S GPUs.
Inference. For inference, all models decode with temperature 0.7 and top-p 0.1, generating up to 1,024 tokens. DeepSeek-R1, Claude Sonnet 4.5, and Nova 2 Lite are served via Amazon Bedrock; Qwen3-8B, Ministral-8B, and our models are run locally. Latency for TTP-R1 is measured on a single NVIDIA L40S GPU with unoptimized HuggingFace inference.

4.4 Evaluation Metrics

We report sample-level Precision, Recall, and F1 over predicted vs. ground-truth technique ID sets, macro-averaged across all test samples. For each test sample, let P_i be the set of predicted technique IDs and G_i the set of ground-truth IDs. We compute:

$$\text{Precision}_i = \frac{|P_i \cap G_i|}{|P_i|},$$
$$\text{Recall}_i = \frac{|P_i \cap G_i|}{|G_i|},$$
$$\text{F}_{1i} = \frac{2 \cdot \text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i},$$

Final metrics are macro-averaged: $\text{F}_1 = \frac{1}{N}\sum_{i=1}^{N} \text{F}_{1i}$. We evaluate at two granularities: technique-level (parent IDs, e.g., T1027) and sub-technique-level (full IDs, e.g., T1027.002). At the sub-technique level, a prediction is correct only if it exactly matches a ground-truth full technique ID. Predicting a parent technique when a sub-technique is the gold label, or vice versa, is not considered correct. At the technique level, all sub-technique IDs are first truncated to their parent IDs (e.g., T1027.002 → T1027) before computing metrics. We also report average F1 across all four datasets and average rank.

4.5 Main Results

Tables 2 and 3 present results at the technique and sub-technique levels, respectively. TTP-R1 achieves the highest average F1 at both granularities and the best average rank across all four datasets. We highlight three key findings.
Non-fine-tuned models over-predict while SFT models under-predict. Non-fine-tuned models tend to over-predict, achieving high recall at the expense of precision. For example, Claude Sonnet 4.5 attains recall above 67% on all datasets yet precision drops as low as 28% on Derived Procedures. It predicts on average 2 labels per sample on single-label benchmarks and 4.2 on multi-label benchmarks (ground-truth average 1.7). Fine-tuned models exhibit the opposite pattern. TechniqueRAG achieves the highest precision among all methods on Expert (75.2%) but recall drops to just 37.7%. It predicts only 1.01 labels per sample on average—including Expert where the ground truth averages 3.32—suggesting that token-level fine-tuning drives the model toward conservative, single-label predictions that miss co-occurring techniques. This motivates our RL stage: rather than optimizing token-level likelihood, which lacks direct control over set-level metrics, decomposed rewards allow the optimizer to recover recall from the conservative SFT policy while preserving its precision gains.
Domain-specific fine-tuning closes the gap between small LMs and frontier models. Without fine-tuning, raw 8B-class models perform poorly, far below frontier LLMs. RAG narrows this gap substantially, with disproportionate gains for smaller models: 8B-class LMs improve by 121–215% in average F1 at the technique level, compared to 8–11% for frontier models. This indicates that a constrained candidate set is most beneficial when the base model lacks domain knowledge, transforming open-ended generation into a manageable selection task. SFT pushes performance further still, surpassing Claude Sonnet 4.5 w/ RAG. This suggests retrieval with structured fine-tuning is more effective than scaling model size alone.
RL improves F1 by selectively boosting recall. RL achieves relative gains of 3.4% at both granularities over the SFT baseline (i.e., TTP-R1 w/o RL), with improvements concentrated on the more challenging multi-label datasets. Recall increases by 18% on Derived

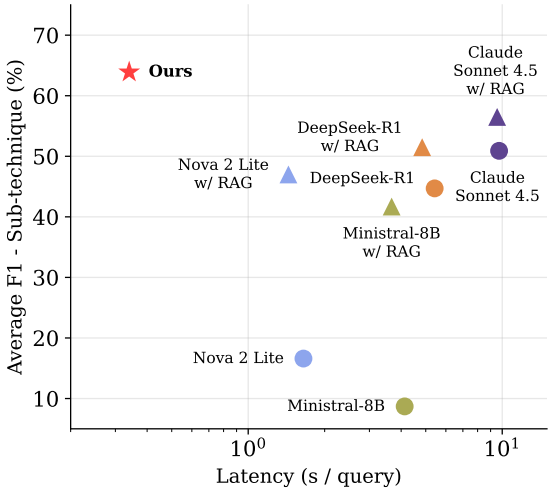

Figure 5: Per-query latency (seconds) vs. average sub-technique F1. TTP-R1 achieves the highest F1 at 0.34s per query—28× faster than Claude Sonnet 4.5.

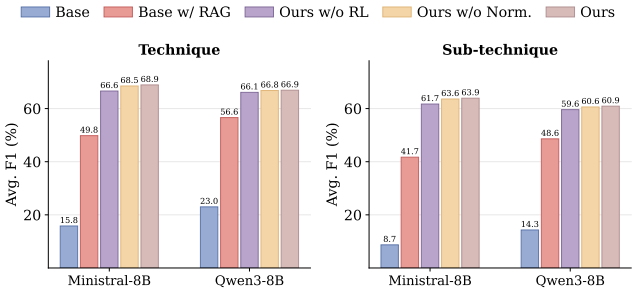

Figure 6: Ablation study across two base models (Ministral-8B and Qwen3-8B). RAG yields the largest gain, SFT the second-largest, and RL adds a targeted boost concentrated on multi-label benchmarks (see Table 2 and 3).

Procedures and 24% on Expert at technique-level, while Procedures dataset remains stably high performance. This comes with a modest precision trade-off, but the net F1 effect is consistently positive—most notably improving by 12% on Expert.

4.6 Efficiency Analysis

Figure 5 shows per-query latency against average sub-technique F1. Baseline latencies are measured via Amazon Bedrock API calls, which benefit from Bedrock's optimized serving infrastructure. TTP-R1 is measured locally on a single L40S GPU with unoptimized HuggingFace inference—a conservative estimate, as production serving (e.g., vLLM) would further reduce latency.

TTP-R1 achieves the highest F1 at 0.34s per query. This is 4× faster than Nova 2 Lite (1.44s), 14× faster than DeepSeek-R1 (4.84s), and 28× faster than Claude Sonnet 4.5 (9.55s). The gap is driven by output length. Reasoning models like DeepSeek-R1 produce long chain-of-thought traces before arriving at an answer, and Claude Sonnet 4.5 generates detailed per-technique justifications, both yielding hundreds of output tokens. Fine-tuning teaches TTP-R1 to directly emit technique indices in ∼9 tokens on average, and since autoregressive decoding scales linearly with sequence length, this brevity translates directly into the observed speedup.

4.7 Ablation Study

Figure 6 presents the ablation study across two base models (Ministral-8B and Qwen3-8B). The largest gain comes from using RAG, which constrains the output space from hundreds of techniques to a manageable candidate list. SFT contributes the second-largest improvement by teaching the model to reliably parse candidates and produce structured outputs. RL adds a smaller but targeted gain, concentrated on multi-label examples (e.g., +5.6 F1 on Expert for Ministral-8B) without degrading easier benchmarks. Removing decoupled reward normalization reduces Ministral-8B by 0.4 F1, suggesting the benefit of it stabilizing training by preventing high-variance reward components from dominating the gradient. Notably, both base models converge to similar final performance, despite Qwen3-8B starting substantially higher in the zero-shot setting. This suggests that the pipeline's effectiveness is driven more by the retrieval and training stages than by the base model's prior knowledge.

5 Conclusion

We presented TTP-R1, a two-stage framework for multi-label attack technique extraction from cyber threat intelligence text. By combining retrieval-augmented supervised fine-tuning with reinforcement learning using decomposed verifiable rewards, TTP-R1 directly optimizes set-level correctness and completeness—addressing a fundamental limitation of token-level training objectives used by prior methods. Across four CTI benchmarks, TTP-R1 achieves state-of-the-art average F1 at both technique and sub-technique granularities, with particularly large gains on recall-demanding multi-label datasets. The approach is efficient: an 8B-parameter model running on a single GPU achieves up to 28× lower latency than frontier LLM baselines while delivering superior average performance. Ablation studies confirm that each component—retrieval, SFT, RL, and decoupled normalization—contributes to the final result, and the pipeline generalizes across two base model families.

Limitations and future work. Our approach has several limitations. First, the retriever imposes an upper bound on recall: if the correct technique is not in the candidate set, the model cannot predict it. Improving retrieval coverage, e.g., through query expansion or iterative retrieval, could address this. Second, the RL gains, while consistent, are modest compared to the SFT stage, suggesting that the current reward design may not fully exploit the potential of policy optimisation. Third, our evaluation is limited to sentence- and paragraph-level extraction; scaling to full-document TTP extraction remains an open challenge due to lack of labeled benchmark. Promising directions include incorporating analyst-in-the-loop feedback to refine the reward signal on ambiguous cases, extending the framework to full-document TTP extraction, and adapting the approach to related or evolving taxonomies (e.g., new ATT&CK versions or MITRE ATLAS [19] for AI system threats) via few-shot updates [32].

References

[1] Basel Abdeen, Ehab Al-Shaer, Anoop Singhal, Latifur Khan, and Kevin Hamlen. 2023. Smet: Semantic mapping of cve to att&ck and its application to cybersecurity. In *IFIP annual conference on data and applications security and privacy*. Springer, 243–260.

[2] Md Tanvirul Alam, Dipkamal Bhusal, Shaswata Ahmad, Nidhi Rastogi, and Philip Worth. 2026. Minerva: Reinforcement Learning with Verifiable Rewards for Cyber Threat Intelligence LLMs. *arXiv preprint arXiv:2602.00513* (2026).

[3] Md Tanvirul Alam, Dipkamal Bhusal, Youngja Park, and Nidhi Rastogi. 2023. Looking beyond iocs: Automatically extracting attack patterns from external cti. In *Proceedings of the 26th international symposium on research in attacks, intrusions and defenses*. 92–108.

[4] Anthropic. 2025. Introducing Claude Sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5. Accessed: 2026.

[5] Marvin Büchel, Tommaso Paladini, Stefano Longari, Michele Carminati, Stefano Zanero, Hodaya Binyamini, Gal Engelberg, Dan Klein, Giancarlo Guizzardi, Marco Caselli, et al. 2025. {SoK}: Automated {TTP} Extraction from {CTI} Reports–Are We There Yet?. In *34th USENIX security symposium (USENIX Security 25)*. 4621–4641.

[6] Center for Threat-Informed Defense. 2023. Threat Report ATT&CK Mapper (TRAM). https://github.com/center-for-threat-informed-defense/tram/. Accessed: 2025.

[7] Andrew Crossman, Jonah Dodd, Viralam Ramamurthy Chaithanya Kumar, Riyaz Mohammed, Andrew R Plummer, Chandra Sekharudu, Deepak Warrier, and Mohammad Yekrangian. 2026. Constructing Multi-label Hierarchical Classification Models for MITRE ATT&CK Text Tagging. *arXiv preprint arXiv:2601.14556* (2026).

[8] Mohamed Amine Ferrag, Fatima Alwahedi, Ammar Battah, Bilel Cherif, Abdechakour Mechri, and Norbert Tihanyi. 2024. Generative ai and large language models for cyber security: All insights you need. *Available at SSRN 4853709* (2024).

[9] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948* (2025).

[10] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=nZeVKeeFYf9

[11] Ghaith Husari, Ehab Al-Shaer, Mohiuddin Ahmed, Bill Chu, and Xi Niu. 2017. Ttpdrill: Automatic and accurate extraction of threat actions from unstructured text of cti sources. In *Proceedings of the 33rd annual computer security applications conference*. 103–115.

[12] Amazon Artificial General Intelligence. 2025. Amazon Nova 2: Multimodal reasoning and generation models. *Amazon Technical Reports* (2025). https://www.amazon.science/publications/amazon-nova-2-multimodal-reasoning-and-generation-models

[13] Udesh Kumarasinghe, Ahmed Lekssays, Husrev Taha Sencar, Sabri Boughorbel, Charitha Elvitigala, and Preslav Nakov. 2024. Semantic Ranking for Automated Adversarial Technique Annotation in Security Text. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*. 49–62.

[14] Ahmed Lekssays, Husrev Taha Sencar, and Tong Yu. 2025. From Text to Actionable Intelligence: Automating STIX Entity and Relationship Extraction. *arXiv preprint arXiv:2507.16576* (2025).

[15] Ahmed Lekssays, Utsav Shukla, Husrev Taha Sencar, and Md Rizwan Parvez. 2025. Techniquerag: Retrieval augmented generation for adversarial technique annotation in cyber threat intelligence text. In *Findings of the Association for Computational Linguistics: ACL 2025*. 20913–20926.

[16] Zhenyuan Li, Jun Zeng, Yan Chen, and Zhenkai Liang. 2022. AttacKG: Constructing technique knowledge graph from cyber threat intelligence reports. In *European symposium on research in computer security*. Springer, 589–609.

[17] Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. 2026. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *arXiv preprint arXiv:2601.05242* (2026).

[18] Mistral AI. 2024. Ministral 8B. https://mistral.ai/news/ministral/. Accessed: 2025.

[19] MITRE Corporation. 2021. MITRE ATLAS: Adversarial Threat Landscape for Artificial-Intelligence Systems. https://atlas.mitre.org/ Accessed: 2026-05-12.

[20] Tu Nguyen, Nedim Šrndić, and Alexander Neth. 2024. Noise contrastive estimation-based matching framework for low-resource security attack pattern recognition. In *Findings of the Association for Computational Linguistics: EACL 2024*. 355–373.

[21] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[22] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*. 3982–3992.

[23] Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Vol. 4. Now Publishers Inc.

[24] Asaf Shabtai et al. 2024. LLMCloudHunter: Harnessing LLMs for Automated Extraction of Detection Rules from Cloud-Based CTI. *arXiv preprint arXiv:2407.05194* (2024).

[25] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300* (2024).

[26] Blake E Strom, Andy Applebaum, Doug P Miller, Kathryn C Nickels, Adam G Pennington, and Cody B Thomas. 2018. Mitre att&ck: Design and philosophy. (2018).

[27] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[28] Junshen Xu, Jiayun Zhang, and Yi Fan. 2026. Multi-Domain Marker Aggregation for Threat Detection in Cloud Environments. In *Proceedings of the ACM Web Conference 2026*. 7809–7820.

[29] Ming Xu, Hongtai Wang, Jiahao Liu, Yun Lin, Chenyang Xu Yingshi Liu, Hoon Wei Lim, and Jin Song Dong. 2024. Intelex: A llm-driven attack-level threat intelligence extraction framework. *arXiv e-prints* (2024), arXiv–2412.

[30] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL] https://arxiv.org/abs/2505.09388

[31] Yizhe You, Jun Jiang, Zhengwei Jiang, Peian Yang, Baoxu Liu, Huamin Feng, Xuren Wang, and Ning Li. 2022. TIM: threat context-enhanced TTP intelligence mining on unstructured threat data. *Cybersecurity* 5, 1 (2022), 3.

[32] Jiayun Zhang, Junshen Xu, Bugra Can, and Yi Fan. 2025. React: Residual-adaptive contextual tuning for fast model adaptation in threat detection. In *Proceedings of the ACM on Web Conference 2025*. 2488–2499.

[33] Shengyao Zhuang, Xueguang Ma, Bevan Koopman, Jimmy Lin, and Guido Zuccon. 2025. Rank-r1: Enhancing reasoning in llm-based document rerankers via reinforcement learning. *arXiv preprint arXiv:2503.06034* (2025).