

CyberPal.AI: Empowering LLMs with Expert-Driven Cybersecurity Instructions

Matan Levi^{1,2}, Yair Allouche¹, Daniel Ohayon¹, Anton Puzanov¹

¹IBM Research

²Ben-Gurion University

matanle@{il.ibm.com, post.bgu.ac.il}
{daniel.ohayon,antonp,yair}@il.ibm.com

Abstract

Large Language Models (LLMs) have significantly advanced natural language processing (NLP), providing versatile capabilities across various applications. However, their application to complex, domain-specific tasks, such as cybersecurity, often faces substantial challenges. In this study, we introduce *SecKnowledge* and *CyberPal.AI* to address these challenges and train security-expert LLMs. *SecKnowledge* is a domain-knowledge-driven cyber-security instruction dataset, meticulously designed using years of accumulated expert knowledge in the domain through a multi-phase generation process. *CyberPal.AI* refers to a family of LLMs fine-tuned using *SecKnowledge*, aimed at building security-specialized LLMs capable of answering and following complex security-related instructions. Additionally, we introduce *SecKnowledge-Eval*, a comprehensive and diverse cyber-security evaluation benchmark, composed of an extensive set of cyber-security tasks we specifically developed to assess LLMs in the field of cyber-security, along with other publicly available security benchmarks. Extensive evaluations demonstrate a significant average improvement of up to 24% over the baseline models, underscoring the benefits of our expert-driven instruction dataset generation process. These findings contribute to the advancement of AI-based cyber-security applications, paving the way for robust security-expert LLMs that can enhance threat-hunting and investigation processes.

1 Introduction

The rapid progress of LLMs offers a wide range of new capabilities that would have been considered unrealistic only a few years ago. LLMs have emerged as disruptive technology in domains ranging from healthcare to finance, changing the way we consume information and perform our daily tasks. As LLMs are trained on trillions of tokens, they should have fundamental knowledge of most domains available online.

One such domain is cyber-security. Yet, cyber-security is also a very complex domain. It requires deep understanding in multiple areas of expertise, such as operating systems, network and communication protocols, malware analysis, threat management, and many others. Furthermore, as cyber-security practice spans from security at the physical layer to security at the application layer, navigating this diverse landscape requires a comprehensive understanding and the abil-

ity to connect disparate elements effectively. Therefore, traditional data generation methods will not be effective (Mitra et al. 2023). As cyber-security is complex and highly domain-expert-driven, it is required to present LLMs with domain-specific data generated from expert knowledge to unlock and harness the potential of LLMs in the field.

Over the past decades, security experts have invested considerable time and resources into monitoring cyber activities, investigating incidents, and producing high-quality reports and comprehensive knowledge bases, which include detection rules for identifying and mitigating threats, among other crucial activities. This study seeks to utilize this extensive domain knowledge and, by integrating it with the capabilities of LLMs, create a highly valuable instruction-tuning dataset that can unlock the potential of LLMs in cyber-security. Overall, we make the following contributions:

- We construct *SecKnowledge*, a reasoning instruction tuning dataset generated using an expert-driven process on a wide range of security-related datasets. The dataset construction involves two main steps. In the first step, instructions are created based on schemas established through domain expertise. These schemas define templates that are filled with domain-expert knowledge and supplemented with LLM-generated content when necessary. In the second step, the initial dataset is expanded through a novel hybrid synthetic content-based data generation process.
- We train *CyberPal.AI*, a family of cyber-security expert LLMs designed to understand and reason about complex security concepts. *CyberPal.AI* demonstrates the advantages of enhancing LLMs with our domain-knowledge instruction dataset, *SecKnowledge*.
- We developed *SecKnowledge-Eval*, a suite of evaluation datasets specifically designed to assess LLMs in the cyber-security domain. *SecKnowledge-Eval* consists of evaluation datasets we constructed to assess LLMs' capabilities on complex cyber-security tasks, alongside public benchmarks, intending to generate a comprehensive and diverse evaluation dataset for assessing both knowledge and understanding of models in the field of cyber-security. *CyberPal.AI* demonstrated superior performance over its baseline models, showing a substantial average improvement of up to 24% in training-aligned

tasks and up to 10% in public cyber-security benchmarks. *CyberPal.AI* also demonstrated better robustness to adversarial benchmarks, showing more than double the robustness of the baseline models.

2 Related Work

General instruction tuning (Wei et al. 2021; Longpre et al. 2023; Raffel et al. 2020; Xu et al. 2022; Sanh et al. 2021; Chung et al. 2022; Ouyang et al. 2022; Chiang et al. 2023) demonstrates how fine-tuning Language Models (LMs) with NLP instructions enhances base models' performance in following general instructions.

Our work falls within the line of research focuses on developing expert LLMs through instruction tuning in specific domains, such as writing assistants (Zhang et al. 2023), arithmetic (Liu and Low 2023), translation (Jiao et al. 2023), medicine (Thawkar et al. 2023), code (Chaudhary 2023; Luo et al. 2023), and many others.

Specifically for the domain of cyber-security, there have been several studies that aimed at training security models. Although not directly related, a line of works trains Encoder-only architecture on security data (Bayer et al. 2022; Park and You 2023; Ranade et al. 2021; Aghaei et al. 2022). However, these models are neither generative nor were trained to follow instructions. Specifically for fine-tuning generative models for cyber-security, VulDetect (Omar and Shiaeles 2023) fine-tuned GPT-2 on a dataset containing both vulnerable and non-vulnerable code. The model is fine-tuned to detect anomalies that represent regular behavior. CyberBench was introduced by Liu, Shi, and Buford (2024) as a cyber-security evaluation dataset that was collected from different works and combined into one security benchmark that includes Name Entity Recognition (NER) tasks for cyber-security corpus, summarization of security blogs, multi-choice Q&A and Classification tasks. SecureFalcon (Ferrag et al. 2023) was trained to differentiate between vulnerable and non-vulnerable C code samples, and is specialized in detecting software vulnerabilities. In contrast to previous efforts, we do not focus on one or more predefined set of tasks. We generate a highly complex and diverse dataset of security instructions spanning a broad spectrum of topics and skills using a domain-expert-driven instruction generation process. As will be described below, we use both domain-expert knowledge alongside LLM generation capabilities to populate our security instruction dataset. This comprehensive dataset enables us to train general-purpose security models.

3 SecKnowledge: Domain-knowledge Driven Cyber-security Instruction Dataset

This section details the construction of *SecKnowledge*, a novel instruction tuning dataset tailored for cyber-security. We leverage expert knowledge and employ a two-step process to build a comprehensive and diverse dataset capable of supporting instruction tuning for various security-related tasks. The two-step process is defined as follows:

1. The first generation step focuses on creating high-quality instructions based on predefined schemas. These

schemas are established through experts-driven in-depth analysis of the diverse set of security datasets, their individual characteristics, and the relationships between different entities within and between datasets. This ensures that the instructions are relevant, accurate, and capture the nuances of various security concepts and tasks. More specifically, each predefined schema consists of rules by which the data-source should be processed into instructions using parsers we developed, ensuring that the generated instructions focus on the important and unique characteristics of the data-source, and are representative of real-world security scenarios. Our method can be considered as an extension to methods such as Wei et al. (2021); Longpre et al. (2023), where templates are simply assigned with predefined questions and answers. In Section 3.1 we break down the generation process.

2. The second generation step expands the generated initial dataset and improves its diversity and complexity. To do so, we employ a hybrid synthetic content-grounded data generation process. More specifically, we fused Evol-Instruct (Xu et al. 2023) and Self-Instruct (Wang et al. 2022) and combined them with content-grounded generation and evaluation pipelines. Additionally, we implemented a routing mechanism between the two generation methods that helps to reduce hallucinations. This process leverages the initial set of instructions from the first generation step to generate additional instructions that follow the established schemas but increase the model's overall generalizability. By incorporating content-grounded synthetic data, we increase the diversity and volume of the final dataset, ultimately leading to more robust and capable security models. In Section 3.2, we further elaborate on the specifics of the generation process.

Our final SecKnowledge dataset consists of various instruction types, among which are: open/closed book question answering, yes/no questions, multi-choice Q&A, CoT (Wei et al. 2022), summarization, logic validation, odd/leave one out multi-choice Q&A, question generation, query/rule explanation and generation, TTP mapping, and others.

Figure 1 illustrates our SecKnowledge generation flow, and Table 1 summarizes the security instruction datasets we composed in the first generation step. We selected these initial data sources because they encompass fundamental security concepts and common, practical scenarios such as rule creation and summarization. We detail the generation process for the primary datasets below. Unless otherwise specified, we use the open-source Mixtral (Jiang et al. 2024) and Mistral-Large-Instruct models for both data generation and evaluation processes.

3.1 First Generation Step: Domain Knowledge-Driven Instruction Generation

Leveraging domain expertise, we first parse and enrich each one of the various security data sources using their unique characteristics and structure, derive connections between the documents in each data-source, and even derive connections between different data sources, as we will describe in the upcoming sections.

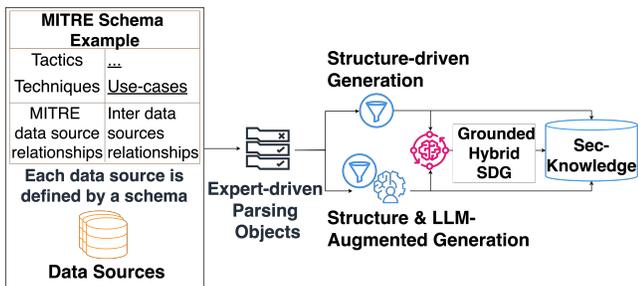


Figure 1: High level overview of our SecKnowledge generation pipeline.

We establish a set of predefined, domain-knowledge-driven, schemas that capture the essential elements of different security tasks. Each schema consists of a series of pre-defined, domain-expertise-driven rules. Each rule is then translated into a parsing object. The parsing object will then generate and fill customized instruction templates with the parsed data. The schemas capture each dataset’s unique objectives and characteristics. This approach ensures that the generated instructions accurately reflect the desired model behavior. The subsequent paragraphs provide a detailed description of the data sources and methodologies employed in the first step of the SecKnowledge data generation process.

Dataset	# of generated instruction
MITRE ATT&CK	45,901
CWE	4,080
CVE	8,447
CAPEC	3,917
Security Wiki	11,000
Security interview Q&A	500
Threat reports	4,500
BRON	62,227
SIEM rules (TTP mapping)	400
Sigma rules	9,329
Security Stack Exchange	2,573
Total	152,874

Table 1: Overview of the initial instructions constructed by the **first step** as described in 3.1. These instructions will be used as the **seed for the second generation step**.

Structure-driven instruction generation The straightforward method for creating an instruction dataset from the documents is to provide a teacher model with raw documents and instruct it to generate instructions based on the content. However, relying on models to produce instructions that simultaneously capture the unique characteristics of datasets while maintaining complexity and diversity proves to be a difficult task. One reason is that models tend to focus on specific or localized sections of a document when generating instructions. More significantly, models struggle to capture and exploit the relationships between different components within each dataset and the relationships between different datasets.

To utilize the structured nature of the different security

datasets and frameworks, we built unique parsers that are fed with the relationships between different entities in the datasets, alongside a schema of required instructions.

Our approach exploits the structured nature of the various cyber-security documents to create a high-quality, diverse, and complex instruction dataset. We demonstrate the efficiency of our method using the MITRE frameworks, among which we can find: MITRE ATT&CK (comprehensive knowledge base of adversary tactics, techniques, detections, and mitigations), CWE (Common Weakness Enumeration), CVE (Common Vulnerabilities and Exposures), CAPEC (Common Attack Pattern Enumeration and Classification), and more. Compiling these frameworks encompasses a vast repository of cyber-security domain knowledge and offers extensive coverage of the security field, making it an excellent resource for fine-tuning our model to the specific requirements and nuances of cyber-security.

Each MITRE framework comprises a structured format that categorizes different aspects of the subject matter, enabling organized analysis of the different security aspects. As such, we create a schema for each MITRE framework, where each framework’s schema defines the following:

1. An instruction set aimed to teach the model the specific characteristics of each object (i.e., tactic, technique, mitigation, detection, attack pattern, etc.) For example, an instruction might detail the relationships between an attack pattern and its corresponding severity, prerequisites, or consequences.
2. An instruction set aimed to teach the model the relationships between different objects *within* each dataset

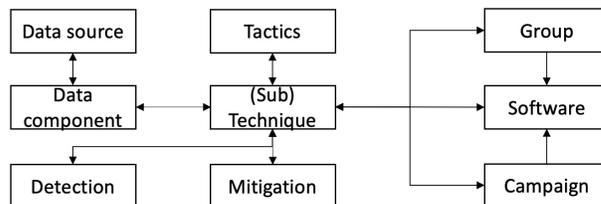


Figure 2: MITRE ATT&CK components relationship.

Next, we provide an example of chain-of-thoughts instruction generated by utilizing the MITRE ATT&CK structure. Figure 2 demonstrates the relationships between different objects within the MITRE ATT&CK framework. Using these relationships, complex instructions are constructed on the wide range of the attack landscape. For instance, see Figure 3, where CoT training example is created from a malicious software to the exploited tactic.

Note that no language model is used during the construction; the connections and relevant text are derived based on our knowledge of the dataset’s structure.

Structured LLM-Augmented Instruction Generation

Previously, we demonstrated how to use raw data alongside domain expertise to populate our schema templates. Here, we combine the same abilities of robust predefined schemas and domain knowledge with the flexibility and rea-

<p>Instruction: Which tactic is related to the following application of software {software_name}: {software_usage}</p> <p>Answer: {software_description}</p> <p>{software_name} is identified by ID {software_id}.</p> <p>The provided usage of {software_name} ({software_id}) pertains to the MITRE sub-technique {sub_technique_name} ({sub_technique_id}). This is since using the given sub-technique, {sub_technique_description}.</p> <p>sub-technique {sub_technique_name} ({sub_technique_id}) is a type of technique {technique_name} ({technique_id}).</p> <p>{technique_name} ({technique_id}) is a type of {tactic_name} ({tactic_id}). Therefore, the answer is: tactic {tactic_name_and_id}.</p>	<p>Instruction: Which tactic is related to the following application of software "Kwampirs": Kwampirs copies itself over network shares to move laterally on a victim network</p> <p>Answer: Kwampirs is a backdoor Trojan used by Orangeworm. It has been found on machines which had software installed for</p> <p>... Kwampirs is identified by MITRE ID S0236.</p> <p>The provided usage of Kwampirs software (S0236) pertains to the MITRE sub-technique: "Remote Services: SMB/Windows Admin Shares" (T1021.002). This is since.....</p> <p>sub-technique "Remote Services: SMB/Windows Admin Shares" (T1021.002) is a type of technique Remote Services (T1021).</p> <p>Technique Remote Services (T1021) is a type of tactic Lateral Movement (TA0008).Therefore, the answer is: tactic Lateral Movement (TA0008).</p>
---	--

Figure 3: Left - a template that maps from malware to tactic. Right - template assignment with the relevant data.

soning abilities of LLMs to create comprehensive instructions. This approach leverages structured templates for consistency while utilizing a teacher model to dynamically generate and fill specific content, ensuring both accuracy and adaptability in instruction creation. More specifically, our main goal of using the teacher model is not to generate general content but rather to harness its reasoning capabilities to guide our models to reason on complex security concepts.

We use this approach on the following datasets: BRON, SIEM Rules to TTP Mapping, and Sigma Rules.

BRON: BRON (Hemberg et al. 2021) is a knowledge graph that interconnects threat data sourced from MITRE ATT&CK, CAPEC, CWE, CVE, MITRE Engage, MITRE D3FEND, MITRE CAR, and exploitdb. This interconnected graph enhances the capabilities of security researchers in conducting advanced threat hunting.

After demonstrating how the MITRE frameworks can be utilized (individually) to generate instructions on the specific characteristics of each MITRE object and the relationships between different objects *within* each framework, we will leverage BRON to generate instructions on the relationships between different objects *across* frameworks.

With hundreds of thousands of nodes and millions of edges interconnecting them, BRON's sheer scale makes it impractical to feed directly to an LLM with the expectation of comprehensively learning all relationships. Therefore, our objective is to generate an instruction set that teaches the model to *reason* if and how different entities are related.

Specifically, using BRON, we have two main goals: 1) construct instructions that will guide LLMs on how to reason if two consecutive entities are related to each other (e.g., CWE and CVE nodes), and 2) showcase the reasoning process for LLMs to derive the path from a specific entity of interest to any other entity in the graph, to accommodate user instruction. This reasoning process will enable a more comprehensive understanding of the relationships between different entities, such as the connection between a platform and its relevant weaknesses, which are not directly related.

To meet the stated goals, the graph should be processed and traversed into paths, which we will later enrich with domain knowledge from the different resources. These paths and explanations will become CoT examples that can guide LLMs to perform effective, complex threat-hunting reasoning. Our processing consists of the following four steps:

Paths Extraction First, we gather all one-step paths between nodes of different types that are directly connected in the graph (e.g., all connections between tactic nodes and technique nodes). Next, for all non-direct paths, we perform a random walk on the graph and construct up to 5000 paths between each pair of node types that are not directly connected (e.g., paths between tactic nodes and CVE nodes).

Derive the Connection Between Direct Nodes After extracting one-step paths between nodes of different types that are directly connected in the graph, we take these direct links and use the reasoning process of a teacher model to explain the connection between each pair of nodes. This involves sending the teacher model instructions that include the descriptions, alongside other information about each node, for each pair of nodes, requiring the teacher model to examine the information and decide if and how the nodes are connected. Additionally, we incorporate negative sampling to illustrate that not all nodes in the graph are connected, compelling the model to make decisions based on the nodes' information. The negative sampling stage is pivotal as it's impractical to present all existing paths (amounting to millions) to the models we fine-tune. Instead, as our models already see these data sources (e.g., CWE) separately during fine-tuning, we aim to equip them with the ability to ascertain whether two nodes are linked based on their information, in the expectation that our models will generalize to paths they haven't encountered during training.

Constructing CoT on Paths Additionally, we present the model with longer paths that involve multiple nodes, between non-direct nodes. For each edge in the path, similar to the direct node processing, we use a teacher model to explain the connection between nodes based on the relationship status (e.g., CVE is a specific implementation of CWE) and the nodes' information (e.g., description). The explanations of all the edges in the path is then chained together to construct a unified CoT explaining that path.

Multi-path CoT Lastly, we construct more complex instructions, i.e., instructions that can be answered only by involving multiple paths from the graph. One example can be a two-stage/paths instruction where we can first ask what the relevant attack patterns for a given weakness are and how to detect/mitigate them in the second stage. See Figure 4 for an example.

Sigma Rules: Sigma is a structured and open signature format that allows to define and describe detection logic. The rule format is flexible and platform-agnostic. The main purpose of Sigma is to provide a structured form in which researchers and analysts can describe their developed detection methods and make them shareable. SigmaHQ¹ is the main rule repository where detection engineers, threat

¹<https://github.com/SigmaHQ/sigma>

Instruction:
Given software epolicy_orchestrator (by mcafee) version 5.10.0, identify a CWE that might be connected to it. Then, suggest some strategies to mitigate it. Explain the rationale behind your choices and how they are relevant to the software.

Answer:
The conceivable link between application epolicy_orchestrator (by mcafee) version 5.10.0 and Authentication Bypass by Alternate Name (CWE-289) is illuminated by the sequence:
application epolicy_orchestrator (by mcafee) version 5.10.0 => CVE-2022-0845 => Authentication Bypass by Alternate Name (CWE-289)

The relationship between application epolicy_orchestrator (by mcafee) version 5.10.0 and Authentication Bypass by Alternate Name (CWE-289) is clarified by the reasoning outlined below:

Step 1:
The CVE-2022-0845 affects the application epolicy_orchestrator version 5.10.0 by McAfee, as it is vulnerable to code injection due to a vulnerability in the pytorchlightning/pytorch-lightning GitHub repository prior to version 1.6.0.

Step 2:
The CWE-289 Authentication Bypass by Alternate Name is being used in CVE-2022-0845 by exploiting the code injection vulnerability in pytorchlightning/pytorch-lightning prior to 1.6.0, where the authentication is performed based on the name of the resource being accessed, but it does not properly check all possible names for that resource.

You can mitigate Authentication Bypass by Alternate Name (CWE-289) by:
1) Avoid making decisions based on names of resources (e.g. files) if those resources can have alternate names.
....

Figure 4: Illustration depicting the construction of a CoT by extracting paths from BRON.

hunters, and all defensive security practitioners collaborate on detection rules. Here, we leverage the repository’s dataset, which contains over 3000 diverse and reliable detection rules as a baseline for our rule instruction set.

Sigma rules contain multiple fields, among which are: the "logsource" field which specifies the type of log data the rule applies to, and the "detection" field which defines the specific conditions that trigger the rule, including event attributes, expected values, and filters for accurate detection. The "level" field indicates the severity of the detected event. Each Sigma rule is connected to the attack it tried to detect.

We take advantage of the Sigma rules structure, feed the relevant fields to a teacher model, and construct the following types of reasoning instruction tasks:

- Step-by-step attack detection explanation using log source and rule detection filters within the detection field.
- Step-by-step reasoning for attack type mapping via detection indicators.
- Sigma rule generation from attack type and/or detection indicators.

We define a schema for each task, ensuring it contains the necessary information. In Figure 5, we showcase the process of converting an entry into step-by-step attack detection instructions. The process works as follows: First, we extract the log source, product, category, detection options, severity level, and any possible rule tags from the given rule. Because Sigma rules are well-structured, we provide the teacher model with an explanation of each field and its role. Next, the teacher model is given the specific values for each field. Finally, it is tasked with explaining the purpose of the rule based on its understanding of these fields and their values. This entire process is fully automated, utilizing a pipeline guided by predefined prompts and a structured flow of actions, making it applicable to any rule. As with other generation processes, we also apply an evaluator that tests the correctness of the generated text.

<p>Sigma rule goal: Detects Obfuscated Powershell via use MSHTA in Scripts</p> <p>logsource: product: windows category: driver_load detection: selection: ImagePath\containsall: 'set', '&&', 'mshta', '(window.close)', 'run', 'vbscript:createobject', condition: selection falsepositives: Unknown level: high tags: attack.defense_evasion, attack.t1027, attack.execution, attack.t1059.001</p>	<p>Instruction: How can I detect Obfuscated Powershell via use MSHTA in Scripts?</p> <p>Answer: To Detect Obfuscated Powershell via use MSHTA in Scripts, you should: Monitor Windows drivers loading for any instances where the ImagePath contains a specific sequence of elements indicating a potential attempt at obfuscation. Specifically, lookout for the simultaneous presence of 'set', '&&', 'mshta', 'vbscript:createobject', 'run', and '(window.close)' within the ImagePath. This detection is critical due to its high severity level ... The relevant tactics and techniques from the MITRE ATT&CK framework associated with this activity include Defense Evasion and Execution, specifically identified as Obfuscated Files or Information (T1027) and Command and Scripting Interpreter: PowerShell (T1059.001).</p>
--	--

Figure 5: Sigma example - the rule is processed into an instruction that explains how to detect a specific attack.

SIEM Rules to TTP Mapping: SIEM (Security Information and Event Management) is a security platform that monitors and correlates threat intelligence, network, and user behavior anomalies to prioritize high-fidelity alerts. We have collected a list of 400 rules from IBM’s SIEM, QRadar, along with their corresponding Tactics, Techniques, and Procedures (TTP) mappings.

TTP mapping of detection rules is critical in cybersecurity as it enables organizations to systematically identify and counteract specific adversary behaviors, thereby enhancing the precision and effectiveness of threat detection.

QRadar’s rules are well-structured and include fields such as rule ID, description, pattern, relevant MITRE tactic/technique ID and name, rule risk level, and more. In the following, we will demonstrate how we leverage this structure to develop a series of instructions for educating CyberPal.AI on mapping rules to Tactics, Techniques, and Procedures (TTPs). Our goal is not merely to create a simple mapping task but to teach the model to *reason* about TTP mapping. To achieve this, we combine expert knowledge with LLMs to generate a comprehensive TTP reasoning instruction dataset, as we describe below.

The process of creating the TTP mapping instruction set involves retrieving the rule description, tactic/technique ID, and name for each rule and its corresponding TTP mapping. Using this information, we access the description and additional relevant data of the tactic/technique from the MITRE ATT&CK framework. We tailor a specific schema, that leverages the required information, and guides the teacher model to reason and clarify the relationship between the rule and the provided TTP based on the descriptions and additional relevant data. Subsequently, the model generates an explanation, which undergoes evaluation for correctness by another model (evaluator). Upon acceptance by the evaluator, a set of instructions is formulated based on the rule, the TTP mapping, and the explanation. The goal of the generated set is to teach *CyberPal.AI* models the expected reasoning process the models need to perform when mapping between rules and TTPs.

Additional Datasets We provided a detailed explanation of the process used to create our primary data sources for the first generation step. Alongside these, we also utilized additional datasets for this first generation step, including security interview Q&A, threat reports on various security threats, discussions from security and reverse-engineering Stack Exchange, and Wikipedia pages focused on computer security. For each such dataset, we define a schema based on its structure and build instructions in a similar manner to the previously mentioned datasets.

3.2 Second Generation Step: Content-Ground Synthetic Data Generation

In the second step of our security instruction generation process, we expand the generated initial dataset from Section 3.1 and improve its diversity and complexity.

We build upon the ideas of Self-Instruct (Wang et al. 2022) and Evol-Instruct (Xu et al. 2023) and fuse them alongside content-grounded generation and an instruction routing mechanism. The reason for switching between the two methods is due to our empirical observations that using In-depth Evol-Instruct with content-grounded generation tends to diverge and generate inaccurate instructions after several iterations (usually around 3), resulting in instructions that an LLM cannot answer, non-grounded instructions, or instructions that deviate from the relevant topic. Therefore, we incorporate a dynamic mechanism that combines Evol-Instruct and Self-Instruct, where in the early stages of the synthetic data generation, we mainly focus on In-depth evolving to generate more complex instructions on the same topic, and as the generation process progresses, we shift the focus towards Self-Instruct, which can be thought of as In-breath evolving, where we mainly focus on generating new tasks, while keeping them grounded and in the same domain as the document. More specifically, the probability of Self-Instruct being chosen is doubled every two iterations (See "Instructions router" in Figure 6). We find that combining Evol and Self-Instruct leads to more grounded instructions due to the difficulty of preserving complex content-grounded instructions in later stages. Using our dynamic method, we managed maximize quality while minimizing errors and increasing quantity. Statistically, when we attempted to generate the same number of instructions using only content-grounded Evol-Instruct, the ratio of hallucinations and incorrect responses was six times higher compared to our method.

Additionally, we incorporate an internal evaluation mechanism using an LLM evaluator. An instruction will be added to the instruction pool and be used in the next iteration only if it passes the evaluator’s assessment. The evaluator is defined by the following objectives:

- Evaluate if the generated instruction is more challenging/complex/rare (for in-depth evolving) or diverse (for Self-Instruct).
- Evaluate if the new instruction is of the same domain as the given instruction based on the document, and evaluate that this new instruction can be answered by the document.

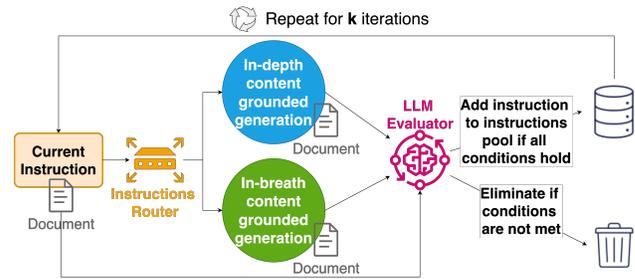


Figure 6: High-level illustration of the content-grounded Synthetic Data Generation (SDG) process.

- Evaluate if the generated answer correctly answers the new instruction, and that the generated answer is grounded by the document.

We employ our content-grounded SDG process to Security Wiki, Security interview Q&A, Security Stack Exchange, MITRE ATT&CK, CWE, CVE, and CAPEC. In the second generation step, we generate an additional 250,000 instructions. The final dataset comprises approximately 400,000 complex and diverse instructions following the two-generation steps.

4 SecKnowledge-Eval: A Comprehensive Security Evaluation Dataset

To assess CyberPal.AI’s performance, we constructed a diverse set of eight new evaluation datasets aimed at testing the model’s capabilities in cyber threat intelligence. To ensure no data contamination between the fine-tuning and testing phases, we partitioned the raw documents into train and test sets, such that the model did not encounter any test-related documents during fine-tuning. After splitting the data, we transformed the documents from the test split into the evaluation tasks described below. Furthermore, we benchmarked CyberPal.AI against another seven public and general cyber-security evaluation datasets to demonstrate its robustness and comprehensive understanding of security concepts. Overall, our evaluation benchmark consists of 15 diverse datasets, with various task types. To the best of our knowledge, this is the most comprehensive cyber-security evaluation benchmark, comprising a diverse set of cyber-security tasks alongside other publicly available security benchmarks. The following paragraphs provide an overview of the evaluation datasets we developed:

4.1 Multiple Choice Tasks

Questions in this section are formatted with four multiple-choice answers, similar to (Hendrycks et al. 2020).

Adversarial MITRE ATT&CK We compiled this dataset using various MITRE ATT&CK sources. This evaluation benchmark is designed to assess the model’s knowledge of malicious software, campaigns, attack tactics, potential detections and mitigations for different attacks, etc. The input consists of information about a given MITRE instance (e.g., description), and the correct answer is the source from

which it was derived (e.g., specific sub-technique). To enhance this evaluation dataset’s difficulty and test our models’ robustness, we developed a novel adversarial attack (Goodfellow, Shlens, and Szegedy 2014; Carlini and Wagner 2017; Levi and Kontorovich 2023) for multi-choice questions on a closed set of options, where the attack chooses the false options (from a closed list of possible options) that will confuse the model with the highest probability, while keeping the original question intact.

Specifically, to increase the difficulty of multiple-choice questions, we developed a novel adversarial attack targeting closed-domain options, where the choices are drawn from a closed list of possible options. Here’s how it works:

1. Given a multiple-choice question where the possible answers are selected from a fixed list of size k , with one of the options being the correct answer.
2. For each of the $k-1$ incorrect options, we create a new classification question. This classification question retains the original question but presents only two options: the correct choice and one of the $k-1$ incorrect options.
3. We then query a language model (LLM) with each of these $k-1$ binary classification questions.
4. From the responses, we identify the incorrect options that the model is most likely to select, given the original question and the correct answer, using the conditional loss on the incorrect option.

The process ensures that the false options selected are those most likely to confuse the model, thereby enhancing the overall difficulty of the dataset without tempering the questions’ content.

Refer to Figure 7 for an example where we posed a question related to one of the MITRE ATT&CK tactics. In this case, one option represents the correct answer, while our method selects the other three options from the list of all possible tactics, prioritizing those most likely to fool an LLM. The attack is an adversarial transfer attack, as we use Phi-3-small as the reference model (the model we attack), and test the attack results using the adversarial generated dataset on the other CyberPal.AI models.

Note that attacking questions related to MITRE ATT&CK tactics requires less computational resources since the list contains only 14 possible options, but on other types of tasks, i.e., technique/software-related tasks, there are hundreds of possible options. Therefore, this attack is time and resource-consuming, but it is done only once, during the generation of the adversarial evaluation dataset.

Regarding the performance degradation of the models on the adversarial dataset: when tested on the Adversarial MITRE evaluation dataset and its non-adversarial version, our fine-tuned CyberPal.AI models demonstrated greater robustness compared to the baseline models, showing more than double the robustness of the baseline models.

Specifically, in Table 2, our models exhibited smaller degradation in performance between the adversarial and non-adversarial versions of the MITRE ATT&CK dataset: for Mistral, our fine-tuned model showed a degradation of 6% in accuracy when tested with the adversarial version of

the dataset, compared to the original Mistral model, which experienced a degradation of 14%. Similarly, for Llama, the fine-tuned model exhibited a 9% degradation in accuracy with the adversarial dataset, in contrast to Llama’s baseline model, which showed a 19% degradation. Finally, for Phi, the fine-tuned model demonstrated a degradation of 7% in accuracy with the adversarial dataset, compared to the Phi base model’s 13% degradation. These results highlight the robust knowledge CyberPal.AI acquired during our fine-tuning process and suggest that CyberPal.AI is more resilient and has successfully generalized to the domain of cyber-security.

Model	ATT&CK Eval. Dataset	
	Original	Adversarial
Mistral-7B-Instruct-v0.3	73.24	59.57 (-13.67)
Sec-Mistral (Ours)	98.87	92.54 (-6.3)
Meta-Llama-3-8B-Instruct	78.59	59.57 (-19.0)
Sec-Llama (Ours)	97.04	87.74 (-9.3)
Phi-3-medium-4k-instruct	77.32	64.50 (-12.8)
Sec-Phi-3-medium (Ours)	96.76	89.57 (-7.1)

Table 2: Models’ results before and after applying our adversarial attack to generate the adversarial multiple-choice dataset. The Original column presents the evaluation dataset results on MITRE ATT&CK prior to the application of our adversarial method. The Adversarial column shows the results on MITRE ATT&CK after applying our adversarial technique. It is evident that CyberPal.AI models exhibit greater robustness to adversarial changes, with their results showing less drastic variation compared to those of the non-security models.

SIEM Rule TTP Mapping SIEM solutions usually include rules that detect a wide range of activities, including excessive firewall denials, multiple failed login attempts, and potential botnet activity. We developed a dataset comprising IBM’s QRadar rules, aiming to classify each rule according to the appropriate tactic or technique.

CTI Detection and Mitigation Mapping As outlined, BRON captures the interrelationships between different Cyber Threat Intelligence (CTI) frameworks. We created a dataset designed to assess model’s proficiency in mapping from tactics, techniques, attack patterns, weaknesses, and vulnerabilities to potential detections and mitigations.

CWE Technical Impact Mapping In CWE, each weakness, if successfully exploited, can lead to one or more technical impacts out of eight options: modify data, read data, DoS: unreliable execution, DoS: resource consumption, execute unauthorized code or commands, gain privileges / assume identity, bypass protection mechanism, and hide activities. This evaluation set presents the model with CWEs and their descriptions, where the goal is to map each CWE to its related technical impact.

4.2 Classification Tasks

CTI Relationship Prediction A major role of our model is to comprehend the relationships between different CTI frameworks. To test this ability, we have built a dataset that

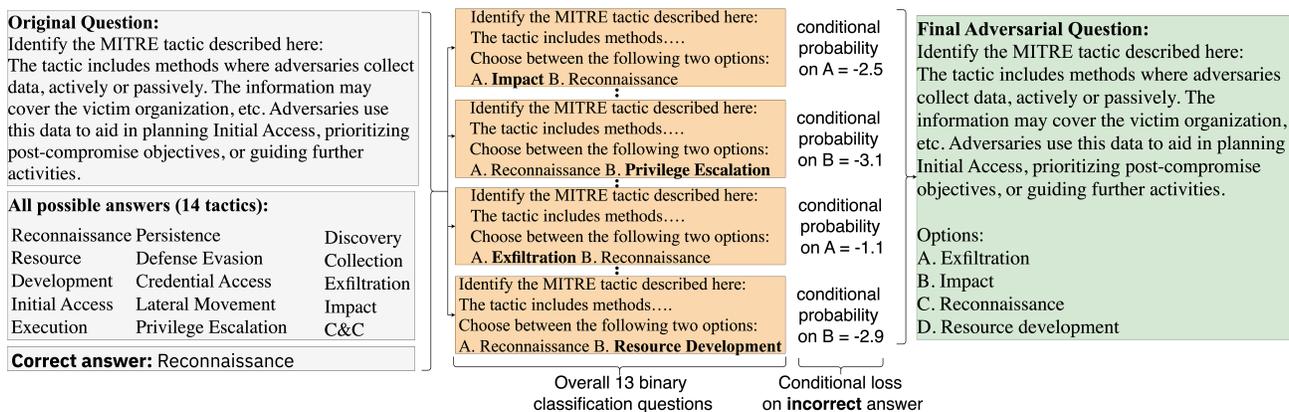


Figure 7: Adversarial MITRE ATT&CK generation pipeline example on a question related to ATT&CK tactics, where there exists 14 possible tactics, one is correct, and we choose three other options that were most likely to fool a third-party LLM.

presents the model with two entities (e.g., instances of CVE and CWE) and two possible explanations—one explaining why the entities are related and another explaining why they aren't. The objective is to determine if the two are related.

CTI Entity Classification This evaluation set consists of various descriptions of CTI entities (such as tactics, malwares, etc.). The objective is to classify whether given description is related to the specified entity.

4.3 Summarization Tasks

CWE Description Summarization We have developed a dataset containing weaknesses from the CWE dataset, intending to summarize the extended descriptions of each CWE. The target of the summarization is the short description provided for each CWE, which aims to offer a concise explanation of the CWE's extended description.

4.4 Public and General Cyber-security Benchmarks

We used the following public multi-choice tasks: CISSP Assessment Questions, MMLU Computer Security (SecMMLU) (Hendrycks et al. 2020), Cybersecurity Skill Assessment², CyberMetric (Tihanyi et al. 2024), Cyber Threat Intelligence Multiple Choice Questions (CTI-MCQ) (Alam et al. 2024), and SecEval (Li et al. 2023). Additionally, we used the Cyber Threat Intelligence Root Cause Mapping (CTI-RCM) classification task introduced in Alam et al. (2024).

5 Experiments

5.1 CyberPal.AI Training Details

Similar to (Mitra et al. 2023), we've empirically noticed that presenting the model with instructions of increasing length improves the model's learning ability. We extend on this idea and employ an incremental training methodology, organized at the dataset level. We structure the datasets into two hierarchical orders: first, we sequence the datasets by their data

²<https://github.com/Ebazhanov/linkedin-skill-assessments-quizzes>

source category, with instructions from simpler data sources introduced first. For example - BRON-related instructions will be presented only after we present the model with the different MITRE frameworks BRON is composed of. In the second hierarchy, within each category, we arrange the instructions based on the increasing length of their outputs.

To train CyberPal.AI, we use our generated *SecKnowledge* dataset. As our base models, we used Llama-3 instruct 8B (AI@Meta 2024), Mistral instruct 7B v0.3 (Jiang et al. 2023), and Phi-3-medium-4k-instruct (Abdin et al. 2024). We employ a learning rate of $4e^{-5}$ for Llama and Phi, and $3e^{-5}$ for mistral. Additionally, we employ linear warm-up for 125 steps. The context length is set to 4096, and effective batch size of 2048 is achieved using gradient accumulation. Based on our empirical findings, beyond 2 epochs, we observed that additional epochs have negligible impact on the final loss before the model starts to overfit.

5.2 Evaluation Metrics

Assessing LLMs on selected datasets requires appropriate evaluation metrics. We apply suitable metrics for each task as described below. For Multiple-choice Q&A, we employ the common HELM (Liang et al. 2022) evaluation method where the token with maximum probability is chosen. For summarization tasks, we use ROUGE (Lin 2004). For summarization, we ran 3 evaluations, computed the average, and removed one standard deviation. Finally, for classification tasks, we use accuracy as the metric. To calculate the average score for evaluation tasks, a straightforward averaging technique is utilized. For summarization tasks specifically, the mean of ROUGE-1, ROUGE-2, and ROUGE-L scores is first determined before calculating the overall average. All tasks except for summarization were tested in greedy mode. All tasks were done in a zero-shot setting.

5.3 CyberPal.AI Results

To demonstrate the effectiveness of our approach, we present SecKnowledge-Eval's results for baseline models and their fine-tuned versions, trained using *SecKnowledge*.

Model	Original/Adv. MITRE ATTACK	SIEM Rule TTP Mapping	CTI Detection and Mitigation	CWE Summarization (R-1/2/L)	Technical Impact Mapping	CTI Relationship Prediction	CTI Entity Classification	Avg.
Mistral-7B-Instruct-v0.3	73.24/59.57	52.05	56.22	28.25/8.16/20.57	59.59	52.44	65.31	52.02
CyberPal.AI-Mistral	98.87/92.54	67.12	70.26	56.78/51.79/54.71	69.05	97.81	83.66	76.41
Meta-Llama-3-8B-Instruct	78.59/59.57	60.27	55.77	26.38/8.16/18.33	59.02	59.76	55.77	52.54
CyberPal.AI-Llama	97.04/87.74	64.38	81.95	46.43/38.45/43.88	66.18	97.81	81.95	74.70
Phi-3-medium-4k-instruct	77.32/64.50	55.47	67.92	27.96/7.83/19.94	66.76	63.75	67.92	57.84
CyberPal.AI-Phi	96.76/89.57	65.07	81.24	48.23/39.24/44.67	68.20	96.27	81.24	75.10

Table 3: Evaluation results for CyberPal.AI models compared to the base model on designated datasets constructed to evaluate the models’ performance on training-aligned security tasks. For the MITRE ATT&CK evaluation set, we provide results for both the original evaluation set and its adversarial version, where we can see that CyberPal.AI demonstrates greater robustness.

Model	CISSP Assessment	SecMMLU	Cybersecurity Skill Assessment	CyberMetric	CTI-MCQ	CTI-RCM	SecEval	Avg.
Mistral-7B-Instruct-v0.3	63.63	67.00	78.69	80.80	58.03	45.85	32.98	60.99
CyberPal.AI-Mistral	89.93	74.00	78.11	81.60	65.33	58.20	42.30	69.92
Meta-Llama-3-8B-Instruct	71.71	74.00	82.24	83.20	63.28	41.45	32.61	64.07
CyberPal.AI-Llama	90.40	77.00	86.98	84.80	66.41	60.65	55.04	74.47
Phi-3-medium-4k-instruct	77.27	78.00	83.43	87.20	65.53	30.68	45.36	66.78
CyberPal.AI-Phi	90.40	80.00	86.39	91.00	72.65	53.00	67.47	77.27

Table 4: Evaluation results for CyberPal.AI models compared to the base on public and general cyber-security benchmarks datasets. Although our models were not trained on these tasks, they exhibit significant and consistent improvement.

In Table 3, we compare the performance of CyberPal.AI across eight training-aligned tasks. Our fine-tuned CyberPal.AI models demonstrate significant and consistent improvements across various tasks, including multiple-choice question answering, summarization, and classification. *Overall, our fine-tuned models achieved a substantial average improvement of 18-24% across all CTI evaluation datasets.* A notable example is the Adversarial MITRE ATT&CK evaluation dataset and its non-adversarial counterpart. As shown in Tables 2 and 3, our fine-tuned models exhibit greater robustness compared to the baseline models. These results highlight CyberPal.AI’s robustness and its ability to generalize effectively to the cyber-security domain.

In Table 4, we compare CyberPal.AI with seven general and public cyber-security benchmarks that the models had not encountered during the fine-tuning process. This evaluation tests whether our fine-tuned models can generalize to tasks different from those they were trained on. *Overall, our fine-tuned models achieved an average improvement of 9-10% across the general cyber-security evaluation datasets.* This is impressive, considering that our models were fine-tuned with different kinds of datasets. See our extended version (Levi et al. 2024) for additional results.

6 Conclusion

In this work, we introduced SecKnowledge, SecKnowledge-Eval, and CyberPal.AI. SecKnowledge is a domain-knowledge-driven cyber-security instruction dataset aimed at fine-tuning LLMs for the security domain. The dataset

is built in two steps: first, we generate instructions based on predefined schemas established through domain expertise; second, we expand this dataset using a hybrid synthetic content-grounded data generation process. CyberPal.AI represents a family of LLMs fine-tuned using SecKnowledge, aimed at developing security-specialized models capable of answering and following complex security-related instructions. To evaluate CyberPal.AI, we introduced SecKnowledge-Eval, a comprehensive suite of evaluation datasets covering diverse cyber-security tasks we developed, alongside publicly available security benchmarks. Our fine-tuned models demonstrated impressive performance on various security-related tasks, including threat hunting (e.g., up 26% in CTI Detection and Mitigation), TTP mapping (up 17% in SIEM Rule Mapping), summarization (up 35% in CWE Summarization), and impact classification (up 11% in CWE technical impact). They also effectively captured relationships between different components within various security frameworks (e.g., up 45% in CTI relationship prediction). CyberPal.AI also showcased enhanced performance on general security knowledge benchmarks such as the security portion of MMLU, skill assessment tests, and analysts’ assessment tests, among others. Overall, CyberPal.AI models outperformed their baseline counterparts, achieving significant average improvement of up to 24% on training-aligned tasks and up to 10% average improvement on public cyber-security benchmarks. These results underscore the extensive knowledge and deep understanding gained through fine-tuning the models with our SecKnowledge dataset.

References

- Abdin, M.; Jacobs, S. A.; Awan, A. A.; Aneja, J.; Awadallah, A.; Awadalla, H.; Bach, N.; Bahree, A.; Bakhtiari, A.; Behl, H.; et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.
- Aghaei, E.; Niu, X.; Shadid, W.; and Al-Shaer, E. 2022. Securebert: A domain-specific language model for cybersecurity. In *International Conference on Security and Privacy in Communication Systems*, 39–56. Springer.
- AI@Meta. 2024. Llama 3 Model Card.
- Alam, M. T.; Bhushl, D.; Nguyen, L.; and Rastogi, N. 2024. CTIBench: A Benchmark for Evaluating LLMs in Cyber Threat Intelligence. *arXiv preprint arXiv:2406.07599*.
- Bayer, M.; Kuehn, P.; Shanehsaz, R.; and Reuter, C. 2022. Cysecbert: A domain-adapted language model for the cybersecurity domain. *arXiv preprint arXiv:2212.02974*.
- Carlini, N.; and Wagner, D. 2017. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 3–14.
- Chaudhary, S. 2023. Code Alpaca: An Instruction-following LLaMA model for code generation. <https://github.com/sahil280114/codealpaca>.
- Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. See <https://vicuna.lmsys.org> (accessed 14 April 2023), 2(3): 6.
- Chung, H. W.; Hou, L.; Longpre, S.; Zoph, B.; Tay, Y.; Fedus, W.; Li, Y.; Wang, X.; Dehghani, M.; Brahma, S.; et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.
- Ferrag, M. A.; Battah, A.; Tihanyi, N.; Debbah, M.; Lestable, T.; and Cordeiro, L. C. 2023. Securefalcon: The next cyber reasoning system for cyber security. *arXiv preprint arXiv:2307.06616*.
- Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- Hemberg, E.; Kelly, J.; Shlapentokh-Rothman, M.; Reinstadler, B.; Xu, K.; Rutar, N.; and O'Reilly, U.-M. 2021. Linking Threat Tactics, Techniques, and Patterns with Defensive Weaknesses, Vulnerabilities and Affected Platform Configurations for Cyber Hunting. *arXiv:2010.00533*.
- Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.
- Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. I.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.
- Jiang, A. Q.; Sablayrolles, A.; Roux, A.; Mensch, A.; Savary, B.; Bamford, C.; Chaplot, D. S.; Casas, D. d. I.; Hanna, E. B.; Bressand, F.; et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.
- Jiao, W.; Huang, J.-t.; Wang, W.; Wang, X.; Shi, S.; and Tu, Z. 2023. Parrot: Translating during chat using large language models. *arXiv preprint arXiv:2304.02426*.
- Levi, M.; Allouche, Y.; Ohayon, D.; and Puzanov, A. 2024. CyberPal. AI: Empowering LLMs with Expert-Driven Cybersecurity Instructions. *arXiv preprint arXiv:2408.09304*.
- Levi, M.; and Kontorovich, A. 2023. Splitting the Difference on Adversarial Training. *arXiv preprint arXiv:2310.02480*.
- Li, G.; Li, Y.; Guannan, W.; Yang, H.; and Yu, Y. 2023. SecEval: A Comprehensive Benchmark for Evaluating Cybersecurity Knowledge of Foundation Models. <https://github.com/XuanwuAI/SecEval>.
- Liang, P.; Bommasani, R.; Lee, T.; Tsipras, D.; Soylu, D.; Yasunaga, M.; Zhang, Y.; Narayanan, D.; Wu, Y.; Kumar, A.; et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.
- Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, 74–81.
- Liu, T.; and Low, B. K. H. 2023. Goat: Fine-tuned llama outperforms gpt-4 on arithmetic tasks. *arXiv preprint arXiv:2305.14201*.
- Liu, Z.; Shi, J.; and Buford, J. F. 2024. Cyberbench: A multi-task benchmark for evaluating large language models in cybersecurity.
- Longpre, S.; Hou, L.; Vu, T.; Webson, A.; Chung, H. W.; Tay, Y.; Zhou, D.; Le, Q. V.; Zoph, B.; Wei, J.; et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, 22631–22648. PMLR.
- Luo, Z.; Xu, C.; Zhao, P.; Sun, Q.; Geng, X.; Hu, W.; Tao, C.; Ma, J.; Lin, Q.; and Jiang, D. 2023. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*.
- Mitra, A.; Del Corro, L.; Mahajan, S.; Coda, A.; Simoes, C.; Agarwal, S.; Chen, X.; Razdaibiedina, A.; Jones, E.; Agarwal, K.; et al. 2023. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*.
- Omar, M.; and Shiaeles, S. 2023. VulDetect: A novel technique for detecting software vulnerabilities using Language Models. In *2023 IEEE International Conference on Cyber Security and Resilience (CSR)*, 105–110. IEEE.
- Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.
- Park, Y.; and You, W. 2023. A Pretrained Language Model for Cyber Threat Intelligence. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 113–122.
- Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text

transformer. *Journal of machine learning research*, 21(140): 1–67.

Ranade, P.; Piplai, A.; Joshi, A.; and Finin, T. 2021. Cyberbert: Contextualized embeddings for the cybersecurity domain. In *2021 IEEE International Conference on Big Data (Big Data)*, 3334–3342. IEEE.

Sanh, V.; Webson, A.; Raffel, C.; Bach, S. H.; Sutawika, L.; Alyafeai, Z.; Chaffin, A.; Stiegler, A.; Scao, T. L.; Raja, A.; et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Thawkar, O.; Shaker, A.; Mullappilly, S. S.; Cholakkal, H.; Anwer, R. M.; Khan, S.; Laaksonen, J.; and Khan, F. S. 2023. Xraygpt: Chest radiographs summarization using medical vision-language models. *arXiv preprint arXiv:2306.07971*.

Tihanyi, N.; Ferrag, M. A.; Jain, R.; and Debbah, M. 2024. CyberMetric: A Benchmark Dataset for Evaluating Large Language Models Knowledge in Cybersecurity. *arXiv preprint arXiv:2402.07688*.

Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N. A.; Khashabi, D.; and Hajishirzi, H. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Wei, J.; Bosma, M.; Zhao, V. Y.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Xu, C.; Sun, Q.; Zheng, K.; Geng, X.; Zhao, P.; Feng, J.; Tao, C.; and Jiang, D. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Xu, H.; Chen, Y.; Du, Y.; Shao, N.; Wang, Y.; Li, H.; and Yang, Z. 2022. ZeroPrompt: scaling prompt-based pretraining to 1,000 tasks improves zero-shot generalization. *arXiv preprint arXiv:2201.06910*.

Zhang, Y.; Cui, L.; Cai, D.; Huang, X.; Fang, T.; and Bi, W. 2023. Multi-task instruction tuning of llama for specific scenarios: A preliminary study on writing assistance. *arXiv preprint arXiv:2305.13225*.