Article

Few-Shot Learning of TTPs Classification Using Large Language Models

Yu Fengrui * and Yanhui Du

Posted Date: 5 January 2024

doi: 10.20944/preprints202401.0372.v1

Keywords: cyber threat intelligence(CTI); TTPs; ATT&CK; data augmentation; large language models(LLMs); supervised fine-tuning(SFT)

Article

Few-Shot Learning of TTPs Classification Using Large Language Models

Fengrui Yu [1,2,*] and Yanhui Du [1]

[1] People's Public Security University of China; enjoy_yfr@163.com
[2] Inner Mongolia Police Professional College; enjoy_yfr@163.com
[*] Correspondence: enjoy_yfr@163.com; Tel.: +86 13451312724

Abstract: Tactics, Techniques, and Procedures (TTPs) constitute the most valuable aspect of Cyber Threat Intelligence (CTI). However, TTPs are often implicit in unstructured text, necessitating manual analysis by field experts. Automating the classification of TTPs from unstructured text is a crucial task in contemporary research. MITRE ATT&CK serves as the de facto standard for studying TTPs. Existing research constructs classification datasets based on its procedural examples for tactics and techniques. However, due to a significant proportion of small sample categories, a long-tail phenomenon exists, leading to a highly imbalanced sample distribution. Consequently, more research concentrates on categories with relatively abundant samples. This paper proposes a method that combines ChatGPT data augmentation with Instruction Supervised Fine-Tuning of open large language models. This approach offers a solution for TTPs classification in few-shot learning scenarios, achieving coverage of 625 technical categories. The Precision, Recall, and F1 scores reach 86.2%, 89.9%, and 87.3%, respectively.

Keywords: cyber threat intelligence(CTI); TTPs; ATT&CK; data augmentation; large language models(LLMs); supervised fine-tuning(SFT)

1. Introduction

The cybersecurity landscape plays a critical, pivotal, and foundational factor influencing national security, societal stability, economic development, and cultural communication. The ongoing cyber conflict in the current digital space is characterized as an "asymmetric" warfare scenario [1]. The information asymmetry between attackers and defenders puts the defensive side in a passive and disadvantaged position. The emergence of Cyber Threat Intelligence (CTI) [2] is gradually improving this situation. CTI stands as a fundamental element in establishing a robust security posture against adversary attacks [3]. It accomplishes this by collecting, analyzing, and identifying real-time cyber threat information through technological exploration, integrating intelligence domain insights with cybersecurity knowledge. This integration effectively curtails the impact and proliferation of cyber attacks.

Following Daviss proposed CTI value and cost measurement system, the "Pyramid of Pain" (PoP [4]) illustrated in Figure 1, TTPs (Tactics, Techniques, and Procedures [5]) represent the highest-value CTI and are simultaneously the most challenging to acquire. TTPs, as advanced Indicators of Compromise (IOCs) [6], comprehensively depict the tactics, techniques, and procedures manifested in attacks or malicious activities. They possess resistance to change and assist security defenders in better understanding the network defense landscape. However, real-world TTPs are often embedded in a vast amount of heterogeneous unstructured text. Relying solely on manual identification requires significant human resources and effort. Additionally, due to the prerequisite of substantial cybersecurity practical experience as knowledge support, this process becomes highly challenging. Consequently, automating the efficient classification of TTPs from unstructured text becomes a crucial task in this research domain.

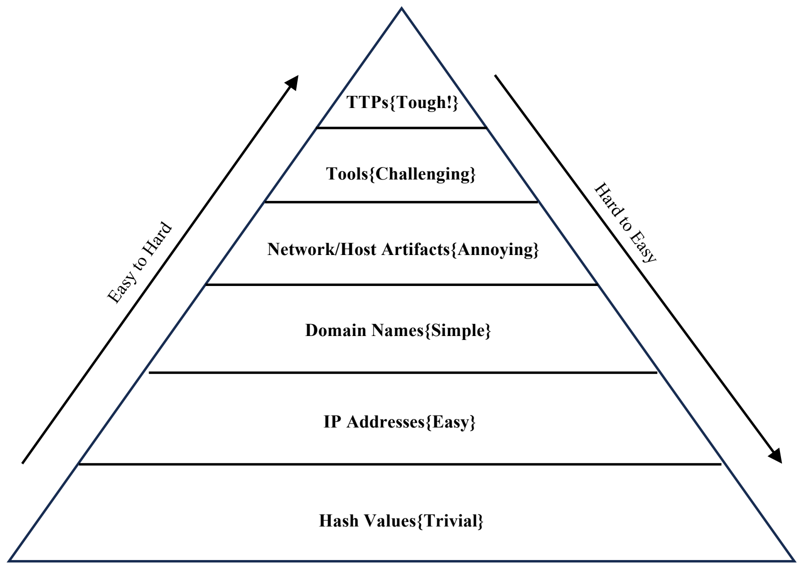


Figure 1. Pyramid of Pain.

Prominent TTPs description frameworks include Stride, Cyber Kill Chain, and MITRE ATT&CK [7]. Among them, ATT&CK is the most renowned, serving as the de facto standard in the field of cyber threat intelligence. ATT&CK V14 encompasses 14 tactics, 201 parent techniques, and 424 sub-techniques. Existing research employs machine learning and deep learning algorithms for text classification, leveraging pre-trained models such as BERT [8] and neural networks. Despite advancements in classification methods and accuracy, several issues persist.

Firstly, concerns arise from the low coverage of technical categories within existing dataset construction methods. Most studies construct datasets based on their specific research tasks, utilizing procedure examples listed in ATT&CK. This method which exhibits a long-tail issue [9] results in a lack of categories for 108 techniques, with some having only descriptions and others having only one procedure example. To address this, some studies remove categories with fewer procedure examples to ensure sufficient training data, leading to low category coverage.

Secondly, for the few-shot TTPs classification task, existing solutions are limited, and classification accuracy is suboptimal. Current research relies solely on data augmentation techniques to address the few-shot learning problem in TTPs classification. However, traditional data augmentation methods prove insufficient to meet the needs of preserving context semantic integrity and enhancing the diversity of training samples. Excessive augmentation can lead to overfitting risks. Additionally, existing deep learning-based classification methods struggle to achieve satisfactory accuracy in few-shot scenarios, encountering performance bottlenecks.

Addressing these challenges, this paper proposes utilizing ChatGPT for data augmentation and leveraging Instruction Supervised Fine-Tuning on large language models to achieve automated TTPs classification in few-shot learning scenarios. The contributions of this research include: 1. Updating existing dataset construction methods to retain all categories with a small number of training samples. Using ChatGPT for data augmentation, the study achieves diverse sample expansion without compromising the original texts contextual semantics. The effectiveness of this research is verified in comparison to other data augmentation methods in the field. 2. Employing Instruction Supervised Fine-Tuning on large language models to solve the few-shot learing of TTPs classification task, achieving full coverage of 625 technical categories with an F1 score of 87.3%, surpassing existing research. 3. Conducting a detailed attribution analysis for categories with low classification recall to assist researchers in clarifying doubts about TTPs classification issues.

2. Related Works

2.1. Data Augmentation Approaches in TTPs Classification

Heejung Kim et al. [10] identified the primary challenge in automating Tactics, Techniques, and Procedures (TTP) classification as the quality of training data, characterized by small datasets and data imbalance. To address this, they proposed applying Synthetic Minority Over-sampling Technique (SMOTE) [11] and Easy Data Augmentation (EDA) [12] techniques for data augmentation, targeting 578 technical categories. Their study demonstrated superior classification performance by leveraging EDA data augmentation, expanding the dataset to 115,030 samples. Another study [13], focusing on 64 technical categories, utilized SMOTE to augment 7,061 existing description sentences to 34,048 samples, achieving a prediction accuracy of 87% and a recall rate of 84%. Additionally, a study [14] proposed using attack trigger words, Indicators of Compromise (IOCs), and description sentences to collectively generate embedded representations. They performed data augmentation on 7 technical categories, each with 700 samples, by systematically replacing text content in sentences with trigger words and IOCs, aiming to extract attack tactics based on description sentences. Existing research commonly employs ATT&CK's procedure examples as benchmark samples, resulting in a severely imbalanced dataset due to varying sentence quantities in the knowledge base and a small dataset size.

While techniques such as SMOTE, EDA, Back Translation, and BERT conditional masking enhance training effectiveness to a certain extent, they face challenges: limited category coverage, excessive augmented samples leading to increased dataset size, and potential loss of semantic integrity and overfitting risks. Notably, these methods generate samples prone to language logic confusion, creating a disconnect between training data and real-world application scenarios. In light of these challenges, leveraging advancements in large language models, such as GPT-3, ChatGPT, and GPT-4, for data augmentation has gained attention.

2.2. Data Augmentation using ChatGPT

Fang et al. [15] utilized GPT-4 for data augmentation to address training data imbalance, discovering that integrating GPT-4 significantly enhances model performance. They validated that the optimal performance improvement occurred with a data generation proportion in the 5%-40%range. Another study [16] employed ChatGPT as a data augmentation method to enhance intent detection task model performance. Addressing the few-shot text classification problem [17], researchers compared data augmentation with meta-learning, prompt-tuning, model design and selected data augmentation which is a concise and efficient approach. They addressed the lack of credibility and diversity in existing augmentation methods by introducing the model AugGPT, utilizing ChatGPT for prompt-based design, resulting in the expansion of samples to 6 and providing a solution for few-shot learning. In line with this research, our study adopts ChatGPT for data augmentation, leveraging prompt construction to enhance sample diversity while preserving the original semantic representations, thus mitigating overfitting risks.

2.3. Model Algorithms for TTPs Classification

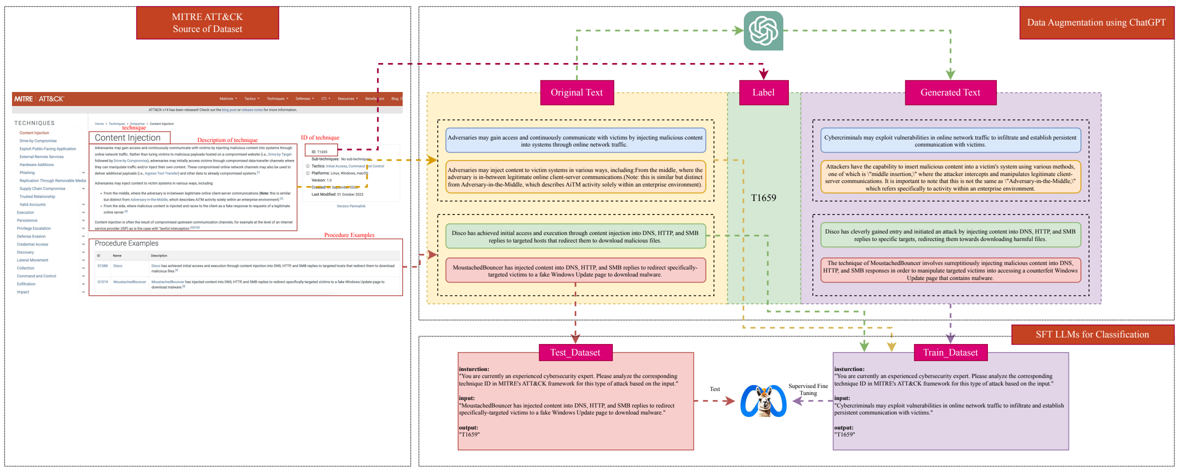
A study [18] introduced the Adaptive Temporal Hierarchical Recurrent Neural Network (ATHRNN) deep learning model, merging cascaded latent variable embeddings with text embeddings, employing attention mechanisms to fuse latent variables related to both text and techniques, ultimately enhancing the classification accuracy of 177 technical categories. Another model [19] defined tactical intelligence recognition and extraction as a multi-classification task. They proposed the Hierarchical Multi-level Attention Convolutional Neural Network (HM-ACNN) deep learning model, utilizing attention mechanisms and convolutional neural networks. By classifying tactics first and transferring weight parameters through knowledge transfer, they fine-tuned the classification of 36 technical categories based on the mapping relationship between tactics and techniques. Addressing the inefficiency and high false positive rates of traditional methods in handling unstructured text classification of tactical intelligence, another study [20] presented the ACRCNN deep learning model. Abstracting the problem into a multi-label classification task, they identified 12 tactics and 36 techniques using BERTs pre-trained model outputs. The study achieved significant performance improvements compared to baseline models through effective feature extraction using bidirectional recurrent neural networks and convolutional neural networks, coupled with attention mechanisms. Yet another model [21], Relation Extraction Network (RENet), utilized the mapping relationship between tactics and techniques to construct an association analysis matrix, enhancing the predictive accuracy for 184 technical categories. Existing studies category numbers struggle to cover half of ATT&CK's technical categories. This limitation stems from the use of small sample datasets and pre-training models, which encounter performance bottlenecks for multi-classification tasks involving 625 technical categories in ATT&CK V14. Researchers have explored solutions involving large language models.

2.4. *Fine-Tuning Large Language Models*

A study [22] proposed training SecureBERT to classify Common Vulnerabilities and Exposures (CVEs) into corresponding technical categories. They highlighted the uncertainty in directly using prompt-based classification with ChatGPT results and emphasized that, in specialized fields like cybersecurity, fine-tuning models for downstream tasks captures subtle details and specific contexts better than generic models. Studies [23][24] confirmed the superior semantic understanding capabilities of large language models. Fine-tuning Large Language Models (LLMs) with small sample datasets showed better performance than traditional deep learning classification methods. Hence, our research employs instruction supervised fine-tuning of open-source large language models to address few-shot learning challenges.

3. Methodology

The entire task process is illustrated in Figure 2. MITRE ATT&CK serves as the data source, with descriptions and procedure examples selected as technical training samples. Employing ChatGPT for data augmentation and Low-Rank Adaptation(LoRA) for Instruction Supervised Fine-Tuning, the study utilizes the LLaMA2-7B-base model to achieve TTPs classification in a few-shot learning scenario.



Adversaries may gain access and continuously communicate with victims by injecting malicious content into systems through online network traffic.

Adversaries may inject content to victim systems in various ways, including:From the middle, where the adversary is in-between legitimate online client-server communications.(Note: this is similar but distinct from Adversary-in-the-Middle, which describes AiTM activity solely within an enterprise environment).

Disco has achieved initial access and execution through content injection into DNS, HTTP, and SMB replies to targeted hosts that redirect them to download malicious files.

MoustachedBouncer has injected content into DNS, HTTP, and SMB replies to redirect specifically-targeted victims to a fake Windows Update page to download malware.

Cybercriminals may exploit vulnerabilities in online network traffic to infiltrate and establish persistent communication with victims.

Attackers have the capability to insert malicious content into a victim's system using various methods, one of which is \"middle insertion,\" where the attacker intercepts and manipulates legitimate client-server communications. It is important to note that this is not the same as \"Adversary-in-the-Middle,\" which refers specifically to activity within an enterprise environment.

Disco has cleverly gained entry and initiated an attack by injecting content into DNS, HTTP, and SMB replies to specific targets, redirecting them towards downloading harmful files.

The technique of MoustachedBouncer involves surreptitiously injecting malicious content into DNS, HTTP, and SMB responses in order to manipulate targeted victims into accessing a counterfeit Windows Update page that contains malware.

insturction:
"You are currently an experienced cybersecurity expert. Please analyze the corresponding technique ID in MITRE's ATT&CK framework for this type of attack based on the input."

input:
"MoustachedBouncer has injected content into DNS, HTTP, and SMB replies to redirect specifically-targeted victims to a fake Windows Update page to download malware."

output:
"T1659"

insturction:
"You are currently an experienced cybersecurity expert. Please analyze the corresponding technique ID in MITRE's ATT&CK framework for this type of attack based on the input."

input:
"Cybercriminals may exploit vulnerabilities in online network traffic to infiltrate and establish persistent communication with victims."

output:
"T1659"

Figure 2. Few-shot learning of TTPs Classification using LLMs.

The following sections first define the mathematical representation of the research task, specifying the models input and output. Subsequently, the critical components of the research, Data Augmentation using ChatGPT and Instruction Supervised Fine-Tuning LLMs for TTPs Classification, are detailed based on the process diagram.

3.1. Mathematical Definition

Our method is represented as follows: Let Σ denote the set of all tokens. The input-output process of the Large Language Model (LLM) can be expressed as $LLM : \Sigma^* \times \Sigma \rightarrow \mathbb{R}$. Assuming the input string is $s \in \Sigma^*$, where s is a technical description statement (e.g., "FunnyDream can send compressed and obfuscated packets to C2."), and the output is $y \in \Sigma$, where y is a technical ID (e.g., "T1659," "T1001"), $LLM(x, y)$ assesses the likelihood of the next token being y, We define this multi-class classification task as $\Sigma^* \rightarrow Y$. Therefore, this task can be directly addressed using LLM, and the formula is:

$$\hat{y} = (\arg\max_{y \in Y} LLM(s, y)) \tag{1}$$

3.2. Data Augmentation using ChatGPT

This study draws on MITRE ATT&CK V14 as the data source, published on October 31, 2023, accessible at https://github.com/mitre-attack/attack-stix-data. The format standard employed is MITRE ATT&CK STIX 2.0 https://oasis-open.github.io/cti-documentation/stix/intro.html. The dataset encompasses 625 categories, comprising 201 parent techniques and 424 sub-techniques. Table 1 details the relevant technical information necessary for the knowledge base, including Technique ID, Description, and Procedure examples.

Table 1. Content of T1659 in MITRE ATT&CK.

Type	Content
Technique ID	T1659
Description	1.Adversaries may gain access and continuously communicate with victims by injecting malicious content into systems through online network traffic. 2.Adversaries may inject content to victim systems in various ways, including: From the middle, where the adversary is in-between legitimate online client-server communications (Note: this is similar but distinct from Adversary-in-the-Middle, which describes AiTM activity solely within an enterprise environment). 3.Content injection is often the result of compromised upstream communication channels, for example at the level of an internet service provider (ISP) as is the case with "lawful interception.
Procedure examples	1.Disco has achieved initial access and execution through content injection into DNS, HTTP, and SMB replies to targeted hosts that redirect them to download malicious files. 2.MoustachedBouncer has injected content into DNS, HTTP, and SMB replies to redirect specifically-targeted victims to a fake Windows Update page to download malware.

Prior methodologies, based on enterprise-attack content in MITREs CTI repository, selectively utilized procedure examples as technical training samples. However, due to the absence of Procedure examples for 108 techniques, such as T1559.003 (Inter-Process Communication: XPC Services), T1547.003 (Boot or Logon Autostart Execution: Time Providers), T1537 (Transfer Data to Cloud Account), etc., which only have technical Descriptions, there arises a need to determine the content for feature descriptions during dataset construction. This dataset utilizes both Procedure examples and technical Descriptions as training samples for each technical category. For longer description texts, a paragraph-based segmentation approach is applied. A total of 14,286 samples are collected, ensuring that each technical category has at least 2 training samples for subsequent dataset preparation.

During processing, a second issue arises: the presence of duplicate procedure examples. For instance, a single procedure example like "APT18 actors leverage legitimate credentials to log into external remote services" involves two techniques, T1078 and T1133. Similarly, another example "OilRig has used credential dumping tools such as LaZagne to steal credentials to accounts logged into the compromised system and to Outlook Web Access" covers five techniques: T1003.004, T1555, T1003.005, T1555.003, and T1552.001. Notably, T1003 is a parent technique named OS Credential Dumping, and T1555 is Credentials from Password Stores. While these techniques have some overlap, T1003 emphasizes accessing credentials through dumping, and T1555 focuses on password storage. These subtle features add complexity to the classification of all technical categories. This dataset adopts a unified method to remove duplicate items, ensuring the diversity and uniqueness of data samples and avoiding model recognition issues during training. A total of 84 categories are affected by duplicate items, resulting in 163 samples. After deduplication, the basic training dataset consists of 14,123 samples.

The distribution of the dataset samples is illustrated in Figure 3 (due to numerous categories, this figure displays 10 categories each with the highest, median, and lowest sample counts, totaling 30 categories).

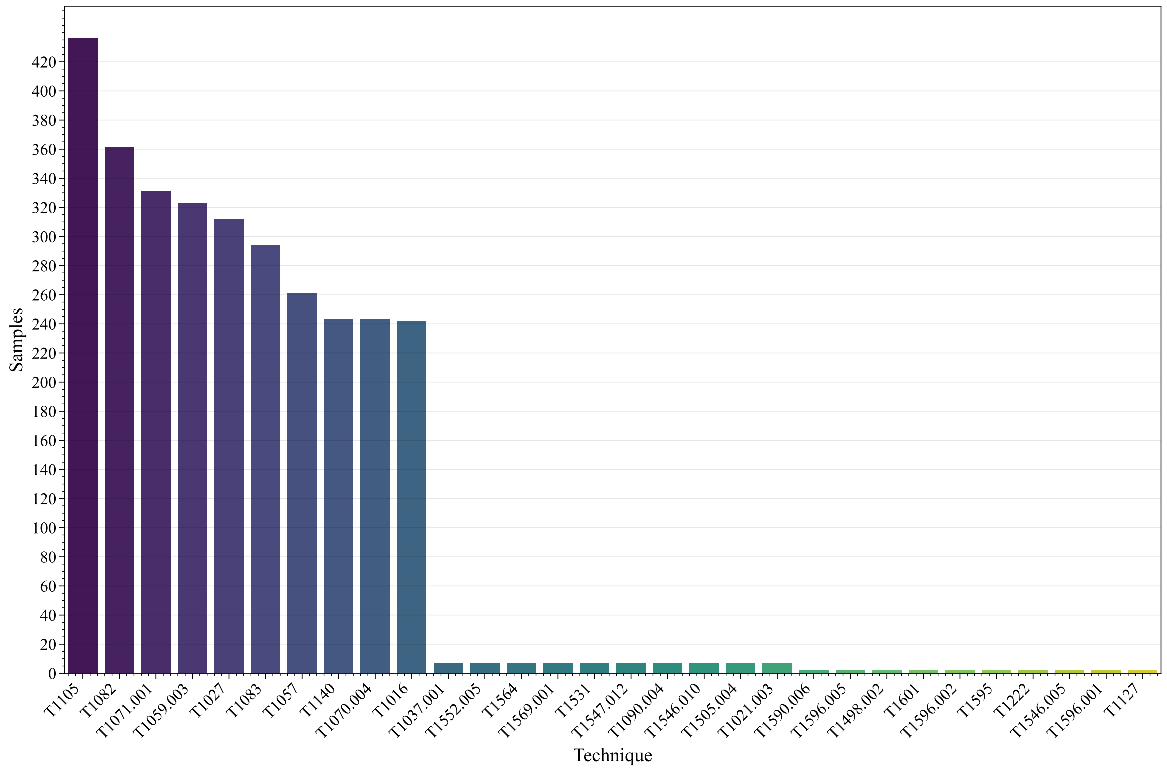


Figure 3. Top, Middle, and Bottom Technique by Sample Count.

It is evident that the distribution of basic training samples is extremely uneven, with the smallest sample quantity being 2 and the largest being 436. Grouping the samples by category and statistically analyzing the sample quantities for different percentiles (50%, 75%, 80%, 85%) reveals sample quantities of 7, 19, 24, and 34, respectively.

This study utilizes ChatGPT for data augmentation [25][26][27], following the pseudocode outlined below. Initially, the 14,123 basic samples are divided into two parts. One sample from each category is randomly selected as an evaluation sample, resulting in 625 samples for the Test dataset and 13,498 samples for the Train dataset. The Train dataset ensures that each category has at least one sample.

The 14,123 data items serve as the base training samples (Base). The content of these base training samples is rewritten using the "gpt-3.5-turbo-instruct" model provided by OpenAI to achieve data augmentation. The generated samples are then combined with the Train dataset to create the training dataset, ensuring that each small-sample category undergoes data augmentation, and the training data does not include the contents of the Test dataset.

The prompt is designed as follows: "As an experienced cybersecurity expert, please creatively rephrase the following statement describing a cyberattack technique: text." The parameters are set to max_tokens = 500, temperature = 0.9. A comparison between the samples generated for the "T1659" category using ChatGPT and the base samples is shown in Table 2.

Table 2. Comparison between original and generated text.

Original Text	Generated Text
1.Adversaries may gain access and continuously communicate with victims by injecting malicious content into systems through online network traffic.	**1.**Cybercriminals may exploit vulnerabilities in online network traffic to infiltrate and establish persistent communication with victims.
2.Adversaries may inject content to victim systems in various ways, including:From the middle, where the adversary is in-between legitimate online client-server communications.(Note: this is similar but distinct from Adversary-in-the-Middle, which describes AiTM activity solely within an enterprise environment).	**2.**Attackers have the capability to insert malicious content into a victims system using various methods, one of which is "middle insertion" where the attacker intercepts and manipulates legitimate client-server communications. It is important to note that this is not the same as "Adversary-in-the-Middle" which refers specifically to activity within an enterprise environment.
3.Disco has achieved initial access and execution through content injection into DNS, HTTP, and SMB replies to targeted hosts that redirect them to download malicious files.	**3.**Disco has cleverly gained entry and initiated an attack by injecting content into DNS, HTTP, and SMB replies to specific targets, redirecting them towards downloading harmful files.
4.MoustachedBouncer has injected content into DNS, HTTP, and SMB replies to redirect specifically-targeted victims to a fake Windows Update page to download malware.	**4.**The technique of MoustachedBouncer involves surreptitiously injecting malicious content into DNS, HTTP, and SMB responses in order to manipulate targeted victims into accessing a counterfeit Windows Update page that contains malware.

The Algorithm 1 for generating samples are as follows: Using the four computed percentiles as benchmarks, if the sample quantity is less than the corresponding benchmark sample quantity, samples are generated to reach that benchmark. The benchmarks are 6, 18, 23, and 33 (compared to the previously mentioned benchmarks, reduced by 1 because one sample is already saved in the Test dataset). For example, if the current categorys sample quantity is less than 18, random samples are chosen and rewritten to achieve the goal of 18 samples. No further increase is made to 50 samples at the 90th percentile. The reason is that for categories with only one base training sample, although data augmentation results in a variety of sample expressions, they essentially convey the same semantic meaning. Thus, generating too many augmented samples may lead to overfitting issues. Therefore, this study adopts only the four mentioned benchmark quantities. According to these rules, a total of four training datasets are generated, with sample quantities of 14,456, 19,427, 21,847, and 27,008.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 January 2024 doi:10.20944/preprints202401.0372.v1

Algorithm 1 Data Augmentation using ChatGPT

```
Require: Base dataset $X_{\text{base}}$, Number of Augmentation $N$    ▷ $N$ is 6, 18, 23, 33
Ensure: Train dataset $X_{\text{train}}$, Test dataset $X_{\text{test}}$
 1: function SPLIT_DATASET($X_{\text{base}}$)
 2:     for $technique$ in $X_{\text{base}}$ do
 3:         $single_string$ = random.choice($technique$)
 4:         $X_{\text{test}}$.append($single_string$)
 5:     end for
 6:     $X_{\text{train}}$ = $X_{\text{base}}$ - $X_{\text{test}}$
 7:     return $X_{\text{train}}$, $X_{\text{test}}$
 8: end function
 9: function AUGMENTATION_CHATGPT($original\_string$)
10:     $response$ = client.completions.create($prompt = prompt + original\_string$)
11:     $generated\_string$ = $response$.choices[0].text
12:     return $generated\_string$
13: end function
14: $X_{\text{train}}$, $X_{\text{test}}$ = SPLIT_DATASET($X_{\text{base}}$)
15: for $technique$ in $X_{\text{train}}$ do
16:     if $technique.value\_count < N$ then
17:         for $i$ in $range(N - technique.value\_count)$ do
18:             $original\_string$ = random.choice($X_{\text{base}}[technique]$)
19:             $generated\_string$ = AUGMENTATION_CHATGPT($original\_string$)
20:             $X_{\text{train}}[technique]$.append($generated\_string$)
21:         end for
22:     end if
23: end for
```

Despite data augmentation, the training samples for each category remain limited, posing a few-shot learning problem. Moreover, considering the large number of categories, it is necessary to combine the understanding capabilities of large language models to achieve Tactics, Techniques, and Procedures (TTPs) Classification.

3.3. Instruction Supervised Fine-Tuning Large Language Models for TTPs Classification

Various high-performance open-source large language models are accessible for research purposes. This study adopts the LLaMA2-7B model provided by Meta [28]. Given the vast number of parameters in large language models, training a model from scratch poses a challenge. The study employs the LoRA method from Parameter-Efficient Fine-Tuning (PEFT) [29] for Instruction Supervised Fine-Tuning. LoRA [30] reduces the number of trainable parameters by learning pairs of rank-decomposition matrices while freezing the original weights. This substantially diminishes the storage requirements for large language models tailored to specific tasks, allowing efficient task-switching during deployment without introducing inference latency. Extensive practical experiments have demonstrated that the LoRA method generally achieves superior fine-tuning results compared to other fine-tuning methods [31].

4. Experiments

4.1. Experimental Setup

The experiments in this study were conducted on a single V100 GPU with 32GB of memory, using Python version 3.8. The training and evaluation parameters are outlined in Table 3.

Table 3. Parameter Settings for Training and Evaluation Processes.

Cutoff length	Learning rate	Epoch	Compute type	Batch_size	LoRA rank	LoRA dropout	Warmup steps	Maxlength	Top-p	Tempreture
1024	5e-4	6	Fp16	8	8	0.1	0	128	0.7	0.95

4.2. Experimental Design

The objectives and specific experimental designs of this study are as follows:

- Demonstrating the Effectiveness of Data Augmentation: In Experiment 1, traditional data augmentation methods such as EDA are compared with the data augmentation method proposed in this study. Both methods are used for Instruction Supervised Fine-Tuning of the LLaMA2-7B model, and the evaluation is performed using the Test dataset. Another data augmentation method, SMOTE, is not selected because it maps inputs to vector representations, which do not match the input of large language models. Previous studies have already demonstrated the superiority of EDA over SMOTE; details can be found in the related work section.
- Demonstrating the Effectiveness of Instruction Supervised Fine-Tuning LLM in Few-Shot Learning: In Experiment 2, baseline models including Baichuan-7B [32], BlueLM-7B [33], ChatGLM3-6B [34][35], Yi-6B(https://huggingface.co/01-ai/Yi-6B) , Bloomz-7B [36], and Qwen-7B [37] are selected. These models are compared with the LLaMA2 model to verify the correctness of choosing LLaMA2 to address the problem in this study. Additionally, baseline models ACRCN and RENet from existing research are chosen to demonstrate the capability of Instruction Supervised Fine-Tuning LLM in solving TTPs classification in a few-shot learning scenario.
- Ablation Experiment: Experiment 3 aims to validate the effectiveness of combining data augmentation and Instruction Supervised Fine-Tuning LLM in solving TTPs classification problems. By selecting the base 14,298 samples as the training dataset, the classification performance of LLM after Instruction Supervised Fine-Tuning is evaluated. Simultaneously, using the dataset based on the benchmark of 33 in data augmentation, ACRCNN and RENet are trained, and the classification performance is evaluated.
- Incorporating the mapping relationship between techniques and tactics in ATT&CK, a detailed analysis was conducted to examine the reasons and countermeasures for low recall in the classification results of technical categories where the model tends to exhibit confusion during the actual prediction process.

4.3. Evaluation Metrics

This section introduces the method used to evaluate the proposed models using our dataset. The experiments use *Precision*, *Recall*, and the $F1$ score from the confusion matrix as performance metrics for the classification models. Since this experiment involves multi-class and imbalanced class problems, emphasis is placed on $macro - F1$ scores as performance metrics. In classification problems, metrics vary based on the number of instances in the target category. Therefore, this study employs macro-average metrics, a method that calculates the average performance of each target category and assesses the performance of classification models with imbalanced categories.

5. Results and Discussion

5.1. Experiment 1

Table 4 and Figure 4 reveal that the use of ChatGPT for data augmentation yields training results comparable to traditional EDA methods on the first two datasets and superior results on the latter two datasets. Both methods show significant performance improvement when the sample size is expanded to 18 instances, with an F1 score increase of nearly 42% compared to six samples. This indicates that within the range of these 18 samples, both methods are effective in data augmentation. However, as the sample size expands from 18 to 33 instances, the training efficiency of EDA methods slows down. While the F1 score increases by 3.5%, the proposed method achieves an F1 score increase of nearly 10%. When the sample size reaches 33 instances, ChatGPT achieves a Recall of almost 90%, a commendable training outcome, whereas EDA methods only achieve 84.5%.

Table 4. Performance Comparison of ChatGPT and EDA Data Augmentation Methods.

Quantity *	**Size**	**Model**	**ChatGPT**			**EDA**		
			Precision	Recall	F1	Precision	Recall	F1
6-50%	14456	LLAMA2	32.7	43.4	35.8	32.8	43.1	35.7
18-75%	19427	LLAMA2	75.6	81.2	77.4	75.6	81.5	77.4
23-80%	21847	LLAMA2	80.2	85.0	81.7	76.4	81.9	78.1
33-85%	27008	LLAMA2	86.2	89.9	87.3	79.4	84.5	80.9

* The meaning of 6-50% in the table indicates categories with fewer than six samples. These categories undergo data augmentation to increase the sample count to six. The 50% signifies that, within the baseline dataset, 50% of the category samples have a count less than six. The same interpretation applies to the other rows.

A clear observation is that ChatGPT demonstrates a better understanding of the context in the underlying training samples, expanding the diversity of sample expressions while maintaining the original semantics. As the number of generated samples increases, ChatGPT's data augmentation method surpasses traditional EDA methods. For instance, given the original sample: 'Footholds to compromised systems may take a variety of forms, such as access to planted backdoors (e.g., Web Shell or established access via External Remote Services In some cases, access brokers will implant compromised systems with a "load" that can be used to install additional malware for paying customers.)', the EDA-generated sample reads: 'web access compromised established may in a variety of forms such as load to planted backdoors e g footholds shell used additional access via external remote services take some cases to brokers will implant compromised systems with systems that access a be or to install can malware for paying customers'. Clearly, the quality, readability, and fluency of samples generated by ChatGPT are superior to those generated by EDA methods. EDA methods involve simple sentence tokenization into word units, with randomly selected units subjected to operations such as deletion, synonym replacement, positional changes, and insertion based on a predetermined ratio. This approach fails to effectively maintain the original semantic meaning during rewriting, and in some cases, it disrupts the original expression, posing the risk of overfitting while lacking sample diversity.

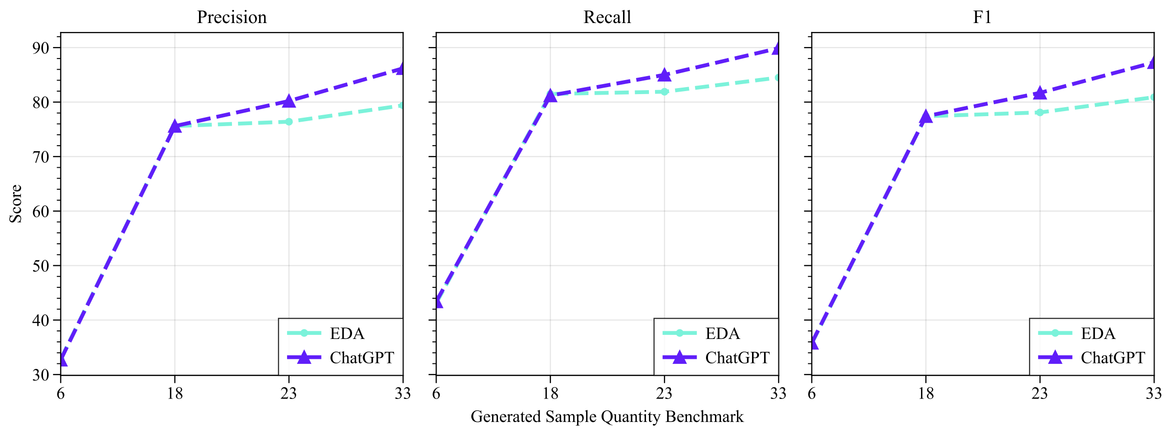


Figure 4. Performance Comparison of ChatGPT and EDA Data Augmentation Methods.

In terms of dataset size, compared to existing research that enhances samples to 115,030 using EDA (refer to related works), the dataset after data augmentation comprises 27,008 samples, a smaller scale. This not only conserves training resources and reduces training time but also achieves higher training efficiency, validating the effectiveness of the proposed data augmentation method.

Based on Table 5, the colored section represents the models predictive performance after data augmentation in corresponding intervals. The most substantial improvement is observed in the category with a sample quantity of "<=6," encompassing 326 categories—the largest category count. F1 score in this interval increases by 85.9%, with sample quantity extended to 33. The "7-17" interval follows with a performance enhancement of 50.8%. The performance change in the "18-22" interval is inconspicuous, exhibiting a 4.6% increase when sample quantity reaches 23, and a marginal 0.4% F1 score improvement when further expanding to 33 samples. Appendix A1 provides detailed statistics on this, elucidating that this trend is associated with the primary technical categories in this interval, namely Privilege Escalation, Defense Evasion, Discovery, and Command and Control, as will be further explained in the analysis of Experiment 4. The "23-32" interval, having only one increase in sample quantity, shows a 7.5% improvement in F1 score. This analysis indicates that the data augmentation method proposed in this study effectively promotes the performance of few-shot learning.

Table 5. Changes in Predictive Accuracy for Each Category Interval as Sample Size Increases

Intervals	**Count**	**Base**			**6**			**18**			**23**			**33**		
		P *	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
<=6	326	9.7	11.8	10.3	**22.2**	**26.3**	**23.5**	**84.9**	**86.0**	**85.3**	**88.3**	**89.1**	**88.6**	**96.1**	**96.4**	**96.2**
7-17	142	29.4	32.8	30.5	27.1	28.4	27.5	**56.0**	**57.1**	**56.4**	**69.6**	**71.3**	**70.1**	**78.0**	**78.9**	**78.3**
18-22	25	32.4	32.4	32.4	38.2	41.2	39.2	48.5	48.5	48.5	**53.1**	**53.1**	**53.1**	**48.4**	**50.0**	**48.9**
23-32	34	58.1	58.1	58.1	52.3	52.3	52.3	47.8	47.8	47.8	54.5	54.5	54.5	**61.9**	**61.9**	**61.9**
>=33	98	67.9	72.6	69.5	59.1	65.7	61.3	66.5	72.4	68.4	64.8	68.5	66.0	66.8	71.9	68.5

* P, R represent Precision, Recall. All the numbers involved in the tables throughout this article are percentages. For example, 22.2 represents Precision=22.2%.

5.2. Experiment 2

As shown in Table 6, this study conducts comparative experiments on fine-tuning multiple open-source large language models using instruction supervision. When the baseline dataset is extended to six samples, the Qwen large language model exhibits the best performance with an F1 score of 51.7%. However, when the sample size reaches 18 instances, the Bloomz model outperforms others, achieving an F1 score of 78.6%. Notably, at sample sizes of 23 and 33 instances, the LLaMA2-7B model demonstrates the highest predictive performance, attaining an F1 score of 87.3% and a Recall of 89.9%. As illustrated in Figure 5, the performance improvement of the LLaMA2 model is the most significant, reaching a 41.6% increase during the transition from six to 18 samples. The experimental results suggest that if the sample size is within 18 instances, one may choose the Qwen or Bloomz models; however, if it exceeds 18 instances, the LLaMA2 model is recommended. This also validates the correctness of selecting the instruction-supervised fine-tuning LLaMA2 model to address the research problem.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 January 2024 doi:10.20944/preprints202401.0372.v1

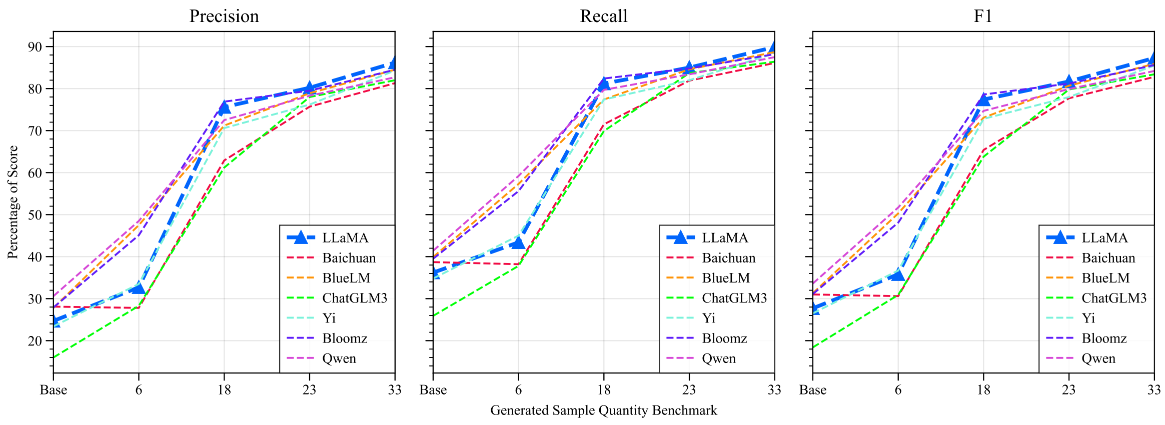


Figure 5. Performance Comparison of Multi LLMs.

In the comparative experiments with the baseline models ACRCNN and RENet, unfortunately, both baseline models exhibit unsatisfactory results when trained on the ChatGPT-33 dataset with the largest number of training samples. The F1 scores are only 1% and 0.4%, respectively. Consequently, experiments on other datasets were not conducted. Both baseline models utilize text embedding based on the pre-trained BERT [8] model, integrating recurrent neural networks and convolutional neural networks to extract temporal and neighborhood features for text classification tasks. However, due to the limited training samples and numerous classification categories, the training results are suboptimal. This further underscores that fine-tuning large language models is a superior solution in the context of few-shot learning classification problems.

5.3. Experiment 3

In the analysis of ablation experiments, the effectiveness of combining the two proposed methods is demonstrated to address the specified issue. Employing a controlled variable approach, we examine the training of baseline models solely using the data augmentation dataset and the fine-tuning of large language models exclusively with the base dataset. Performance analysis in Table 6 reveals that the baseline method exhibits suboptimal few-shot learning performance. Similarly, when fine-tuning is conducted solely with the base dataset, the Qwen-7B model emerges as the top performer, achieving a modest F1 score of 33.6%.

Furthermore, we aim to establish the performance disparity between instruction-supervised fine-tuning and direct model prediction. For direct prediction, multiple preliminary tests are required to construct a suitable prompt. The finalized prompt is as follows: "You are currently an experienced cybersecurity expert. Please analyze the corresponding technique ID in MITREs ATT&CK framework for this type of attack based on the input. Please note that responses must follow this format, such as T1055, T1011, T1055.011, TT1001.003, etc. Each answer should only provide one ID without the need for explanations, and no additional content should be generated. The input is:".Referencing Table 7, it is essential to note that the models utilized in this study are all base versions, such as LLaMA2-7B-Base. There is a distinction in conversational performance between LLaMA2-7B-Base and LLaMA2-7B-Chat. Additionally, ChatGPT is chosen as a comparative model, and the same prompt is employed for predictions. The results demonstrate a substantial enhancement in predictive performance through fine-tuning, as the base versions of models struggle to correctly label inputs. ChatGPT achieves a mere 2.4% F1 score, emphasizing the necessity of fine-tuning.

Table 6. Overview of Model Performance Comparison Based on Dataset Training

Approach	Model	Size	Dataset														
			Base			ChatGPT-6			ChatGPT-18			ChatGPT-23			ChatGPT-33		
			P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SFT	LLaMA2	7B	24.7	36.2	27.7	32.7	43.4	35.8	75.6	81.2	77.4	**80.2**	**85.0**	**81.7**	**86.2**	**89.9**	**87.3**
	Baichuan	7B	28.1	38.7	31.0	27.8	38.2	30.6	62.9	71.5	65.4	75.7	81.9	77.7	81.3	86.1	82.8
	BlueLM	7B	27.8	39.9	31.3	47.5	57.3	50.3	71.2	77.4	73.1	78.9	84.6	80.7	84.5	88.7	85.9
	ChatGLM3	6B	16.0	25.9	18.4	28.3	37.8	30.9	61.2	69.9	63.8	78.0	83.7	79.8	82.0	86.4	83.4
	Yi	6B	23.4	34.8	26.5	33.5	45.0	36.7	70.6	77.4	72.8	76.2	82.0	78.0	84.3	88.4	85.5
	Bloomz	7B	27.9	39.5	31.1	45.1	55.6	48.1	**76.9**	**82.4**	**78.6**	79.5	84.8	81.2	84.5	88.2	85.6
	Qwen	7B	**30.6**	**41.4**	**33.6**	**48.5**	**59.2**	**51.7**	72.5	79.7	74.7	78.3	83.4	79.9	82.7	87.5	84.2
Baseline	ACRCNN	458M	-	-	-	-	-	-	-	-	-	-	-	-	0	2.1	1.2
	RENet	431M	-	-	-	-	-	-	-	-	-	-	-	-	0	0	0

In conclusion, the ablation experiment analysis supports the conclusion that, for the few-shot learning scenario addressed in this study regarding TTPs classification, the combination of instruction-supervised fine-tuning of large language models and the use of ChatGPT for data augmentation provides an optimal solution.

Table 7. Performance Comparison of Instruction Supervised Fine-Tuning versus Base

Models	SFT			Base		
	P	R	F1	P	R	F1
LLaMA2	86.2	89.9	87.3	0	0	0
Qwen	82.7	87.5	84.2	0	0	0
ChatGPT	-	-	-	1.9	4.6	2.4

5.4. Experiment 4

Based on the mapping relationship between the 14 tactics and 625 techniques in ATT&CK, this experiment provides a detailed analysis of categories with low recall during the models prediction process. The distribution of predictions for the 14 categories is presented in the table 8. The best-performing tactic is Persistence, achieving Precision, Recall, and F1 values of 97.4%, 98.3%, and 97.7%, respectively. This tactic includes 115 technical categories, and based on the aforementioned five intervals, the number of categories in each interval is 74, 28, 3, 2, and 8.

The tactic with the poorest predictive performance is Command and Control, with Precision, Recall, and F1 values of 51.5%, 59.5%, and 53.8%, respectively. It encompasses 40 tactic categories, with the number of categories in each interval being 8, 10, 4, 6, and 12. In comparison to Execution (36 items) and Discovery (46 items), which have similar category distributions but significantly better performance, it is evident that the poor performance is not attributed to data augmentation but rather to inherent confusion in the technical descriptions themselves.

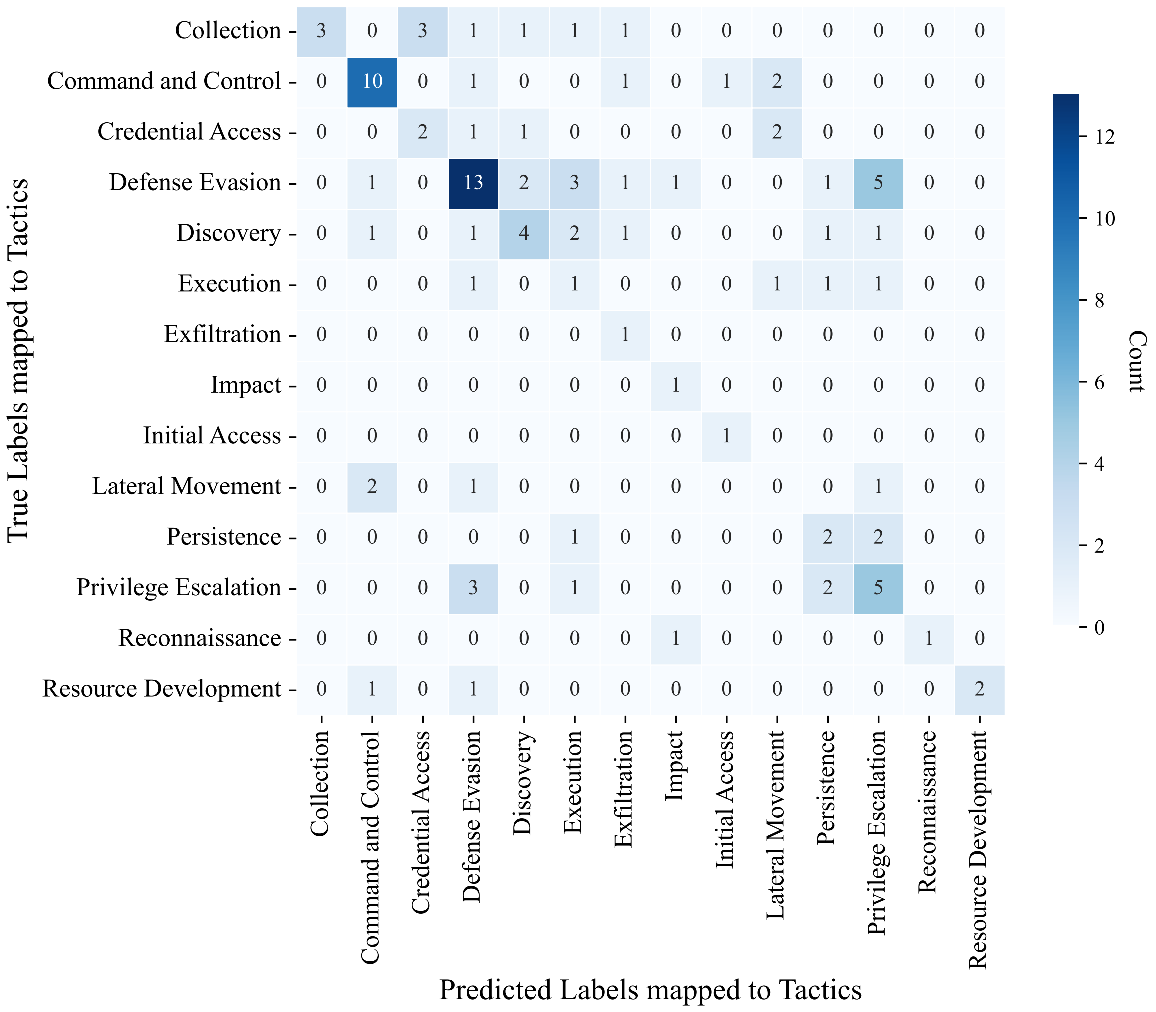
In a further experimental analysis, detailed in Appendix A2 A3 A4 A5 A6, we compiled information on the 30 technical categories consistently mispredicted by the five LLaMA2-7B models after instruction-supervised fine-tuning using both the baseline dataset and the data-augmented dataset in each interval. These categories span Command and Control (7 items), Defense Evasion (7 items), Discovery (5 items), Collection (4 items), Resource Development (3 items), and other tactics encompassing a total of 11 items. Its important to note that some technical categories belong to multiple tactics, resulting in overlapping statistics in the aforementioned counts.

Table 8. Performance Comparison of Technical Category Predictions Based on Tactics

Reconnaissance			Resource Development			Initial Access			Execution			Persistence			Privilege Escalation			Defense Evasion		
P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
92.2	93.3	92.6	85.1	87.2	85.8	92.9	95.2	93.7	85.5	86.8	85.9	97.4	98.3	97.7	93.1	95.2	93.8	84.9	87.3	85.6
Credential Access			**Discovery**			**Lateral Movement**			**Collection**			**Command and Control**			**Exfiltration**			**Impact**		
P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
89.4	90.9	89.9	73.0	76.0	74.0	76.9	76.9	76.9	65.5	69.0	66.7	51.5	59.1	53.8	92.1	94.7	92.9	94.4	96.3	95.1

The predicted technical categories belonging to a common parent technique total 5 items, namely T1564, T1195, T1053, T1074, and T1055. For example, the true label T1564 corresponds to the technique named "Hide Artifacts," with a partial description excerpt: "Adversaries may attempt to hide artifacts associated with their behaviors to evade detection." The predicted label is T1564.001, associated with the technique "Hide Artifacts: Hidden Files and Directories," and its partial description is: "Adversaries may set files and directories to be hidden to evade detection mechanisms." There is an inherent hierarchical relationship between these labels, with overlapping content in their descriptions. In a test statement from the test set, such as "Tarrask is able to create hidden scheduled tasks by deleting the Security Descriptor (SD) registry value," the models attention may be more focused on the term "hidden," leading to a prediction error. However, in practical scenarios, the models ability to predict the specific parent technique accurately is considered commendable performance.

The predicted technical categories falling under a common tactic amount to 14 items, including 4 techniques for Command and Control, 3 for Defense Evasion, and 2 for Discovery. Taking the Discovery tactic as an example, one technical category is T1069.003, "Cloud Groups," with training data excerpt: "ROADTools can enumerate Azure AD groups." The predicted technical category is T1087.004, "Cloud Account," with training data excerpt: "APT29 has conducted enumeration of Azure AD accounts." In the test dataset: "During C0027 Scattered Spider accessed Azure AD to download bulk lists of group members and their Active Directory attributes," both training instances share overlapping content, specifically "Azure AD," despite the distinction between "group" and "account." This leads to prediction errors in the model. For future training iterations, focusing on fine-tuning for details can enhance the models predictive capabilities.



	Collection	Command and Control	Credential Access	Defense Evasion	Discovery	Execution	Exfiltration	Impact	Initial Access	Lateral Movement	Persistence	Privilege Escalation	Reconnaissance	Resource Development
Collection	3	0	3	1	1	1	1	0	0	0	0	0	0	0
Command and Control	0	10	0	1	0	0	1	0	1	2	0	0	0	0
Credential Access	0	0	2	1	1	0	0	0	0	2	0	0	0	0
Defense Evasion	0	1	0	13	2	3	1	1	0	0	1	5	0	0
Discovery	0	1	0	1	4	2	1	0	0	0	1	1	0	0
Execution	0	0	0	1	0	1	0	0	0	1	1	1	0	0
Exfiltration	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Impact	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Initial Access	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Lateral Movement	0	2	0	1	0	0	0	0	0	0	0	1	0	0
Persistence	0	0	0	0	0	1	0	0	0	0	2	2	0	0
Privilege Escalation	0	0	0	3	0	1	0	0	0	0	2	5	0	0
Reconnaissance	0	0	0	0	0	0	0	1	0	0	0	0	1	0
Resource Development	0	1	0	1	0	0	0	0	0	0	0	0	0	2

Figure 6. Confusion Matrix of Techniques mapped to Tactics

Among the 11 technical categories predicted not to belong to the same parent technique or the same tactic, there are 2 related to the Defense Evasion tactic, 2 to Resource Development, 3 to Command and Control, 1 to Lateral Movement, 2 to Discovery, 1 to Execution, and 2 to Collection. Taking Command and Control as an example, the true labels are T1001.001, T1071.002, and T1132.001, while the predicted labels are T1027, T1021.002, and T1048.003, associated with the tactics Defense Evasion, Lateral Movement, and Exfiltration, respectively.

For T1001.001 (Junk Data), the test statement in the test dataset is "PLEAD samples were found to be highly obfuscated with junk code." However, T1027s technical name is "Obfuscated Files or Information." Since T1027 has 327 training samples in the dataset compared to T1001.001s 33 samples, the model leans more toward learning the term "Obfuscated" and overlooks "junk," leading to a prediction error.

Regarding T1071.002 (File Transfer Protocols) and T1021.002 (SMB/Windows Admin Shares), both involve the SMB network protocol. Consequently, the model encounters ambiguity in the prediction process due to the shared element of the SMB protocol in these categories.

Regarding the mispredicted technical categories, the challenges mentioned in reference [38] are noteworthy. The detailed classification of techniques and tactics in ATT&CK can be intricate, and many tactics often occur concurrently. Human analysis can also face ambiguity in distinguishing these situations. The illustration 6 shows the mapping of predicted technical categories to tactical categories after fine-tuning LLaMA2-7B on the ChatGPT-33 dataset.

In the context of Lateral Movement, Defense Evasion, and Command and Control tactics, there are several areas prone to confusion. Frequently, during a cyber attack process, a single technique may serve multiple purposes, achieving Defense Evasion during Lateral Movement, and concurrently accomplishing Privilege Escalation. This undoubtedly adds complexity to predictions and underscores the importance of addressing this during model training. Specifically, it suggests the need to augment training samples for such related techniques and emphasizes careful observation of predictions for these interconnected technical categories during the prediction task.

6. Conclusion & Future Work

This study addresses the challenges of low category coverage and small sample learning in existing TTPs classification problems. The proposed approach, combining ChatGPT for data augmentation and leveraging Instruction Supervised Fine-Tuning on the large language model LLaMA2-7B-base, achieves automated classification of TTPs in few-shot learning scenarios. Experimental validation demonstrates the effectiveness of combining these two methods to address the current challenges in TTPs classification research.

This research represents the first attempt in this domain to use Instruction Supervised Fine-Tuning on large language models to solve downstream tasks. While promising results have been achieved, there is room for further improvement. For TTPs classification problems, future work can focus on:

- Expanding Data Sources: Broaden the range of data sources while ensuring data quality. Increase the quantity and diversity of basic training samples as much as possible.
- Mapping Categories to Numeric Values: Implement a mapping relationship between categories and short strings, such as mapping "T1659" to the number "1" and all 625 technical categories to numbers 1-625. This approach may reduce the generation pressure on large language models, potentially improving classification performance.
- Incorporating Tactics and Techniques: Extend the study to include tactics and techniques, creating an automated one-stop pipeline for classifying TTPs based on unstructured text.
- Introducing Multimodal Knowledge: Current research primarily focuses on the identification and extraction of TTPs from unstructured text. However, intelligence sources encompass various modalities, including text, images, audio, video, etc. Further exploration and research are needed to extend the scope of intelligence recognition and extraction using techniques of Multimodal Large Language Models(MLLM) [39][40][41][42].

The proposed methodology lays the groundwork for advancing TTPs classification research and can be extended to address more complex challenges in the field.

Author Contributions: Conceptualization, Fengrui Yu; methodology, Fengrui Yu and Yanhui Du; validation, Fengrui Yu; data curation, Fengrui Yu; writing—original draft preparation, Fengrui Yu.; writing—review and editing, Yanhui Du; visualization, Fengrui Yu; supervision, Yanhui Du; funding acquisition, Yanhui Du. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Innovative Research Project of Double First-Class in Cybersecurity Law Enforcement Technology at the Peoples Public Security University of China (2023SYL07) and the Key Research Project at the Inner Mongolia Police Vocational College (NMJY2022-LX-ZD007).

Data Availability Statement: The base dataset is sourced from MITREs ATT&CK https://github.com/mitre-attack/attack-stix-data, and all experimental datasets and codes are open-sourced in the GitHub project https://github.com/Yu000910/Few-Shot-Learning-of-TTPs-Classification-Using-Large-Language-Models.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A.

Table A1. Distribution of Related Technical Categories for 14 Tactics in Various Sample Size Intervals

Tactics and Intervals	totle	<=6	7-17	18-22	23-32	>=33
Reconnaissance	44	38	5	1	0	0
Resource Development	45	30	9	2	2	2
Initial Access	21	8	5	1	1	6
Execution	36	15	5	1	2	13
Persistence	115	74	28	3	2	8
Privilege Escalation	104	60	27	5	1	11
Defense Evasion	191	111	40	6	8	26
Credential Access	64	33	20	5	3	3
Discovery	46	10	9	4	4	19
Lateral Movement	23	8	9	1	2	3
Collection	37	12	13	0	2	10
Command and Control	40	8	10	4	6	12
Exfiltration	19	10	4	2	2	1
Impact	27	18	5	0	1	3

Table A2. Statistical Summary of Mispredicted Technical Categories for All Fine-Tuned Models -1

Intervals	Count	True_Label			Predict_Label		
		ID	Tactic	Technique	ID	Tactic	Technique
<=6	3	T1564	Defense Evasion	Hide Artifacts	T1564.001	Defense Evasion	Hide Artifacts: Hidden Files and Directories
		T1102.003	Command and Control	One-Way Communication	T1105	Command and Control	Ingress Tool Transfer
		T1195.001	Initial Access	Compromise Software Dependencies and Development Tools	T1195.002	Initial Access	Compromise Software Supply Chain

Table A3. Statistical Summary of Mispredicted Technical Categories for All Fine-Tuned Models -2

Intervals	Count	True_Label			Predict_Label		
		ID	Tactic	Technique	ID	Tactic	Technique
7-17	8	T1104	Command and Control	Multi-Stage Channels	T1105	Command and Control	Ingress Tool Transfer
		T1069.003	Discovery	Cloud Groups	T1087.004	Discovery	Cloud Account
		T1027.007	Defense Evasion	Dynamic API Resolution	T1106	Execution	Native API
		T1588.001	Resource Development	Malware	T1036	Defense Evasion	Masquerading
		T1053.002	Execution, Persistence, Privilege Escalation	Scheduled Task/Job: At	T1053.005	Execution, Persistence, Privilege Escalation	Scheduled Task
		T1608.004	Resource Development	Drive-by Target	T1583.001	Resource Development	Domains
		T1074	Collection	Data Staged	T1074.002	Collection	Remote Data Staging
		T1001.001	Command and Control	Junk Data	T1027	Defense Evasion	Obfuscated Files or Information

Table A4. Statistical Summary of Mispredicted Technical Categories for All Fine-Tuned Models -3

Intervals	Count	True_Label			Predict_Label		
		ID	Tactic	Technique	ID	Tactic	Technique
18-22	3	T1550.002	Defense Evasion, Lateral Movement	Pass the Hash	T1134.002	Privilege Escalation, Defense Evasion	Create Process with Token
		T1016.001	Discovery	Internet Connection Discovery	T1071.004	Command and Control	DNS
		T1497	Defense Evasion, Discovery	Virtualization/ Sandbox Evasion	T1029	Exfiltration	Scheduled Transfer

Table A5. Statistical Summary of Mispredicted Technical Categories for All Fine-Tuned Models -4

Intervals	Count	True_Label			Predict_Label		
		ID	Tactic	Technique	ID	Tactic	Technique
23-32	5	T1587.001	Resource Development	Malware	T1102	Command and Control	Web Service
		T1572	Command and Control	Protocol Tunneling	T1095	Command and Control	Non-Application Layer Protocol
		T1071.002	Command and Control	File Transfer Protocols	T1021.002	Lateral Movement	SMB/Windows Admin Shares
		T1014	Defense Evasion	Rootkit	T1574.006	Persistence, Privilege Escalation, Defense Evasion	Dynamic Linker Hijacking
		T1027.011	Defense Evasion	Fileless Storage	T1112	Defense Evasion	Modify Registry

Table A6. Statistical Summary of Mispredicted Technical Categories for All Fine-Tuned Models -5

Intervals	Count	True_Label			Predict_Label		
		ID	Tactic	Technique	ID	Tactic	Technique
>=33	11	T1124	Discovery	System Time Discovery	T1053.005	Execution, Persistence, Privilege Escalation	Scheduled Task
		T1203	Execution	Exploitation for Client Execution	T1211	Defense Evasion	Exploitation for Defense Evasion
		T1119	Collection	Automated Collection	T1020	Exfiltration	Automated Exfiltration
		T1056.001	Credential Access, Collection	Keylogging	T1555.005	Credential Access	Password Managers
		T1132.001	Command and Control	Standard Encoding	T1048.003	Exfiltration	Exfiltration Over Unencrypted Non-C2 Protocol
		T1055.001	Privilege Escalation, Defense Evasion	Dynamic-link Library Injection	T1055	Privilege Escalation, Defense Evasion	Process Injection
		T1560.003	Collection	Archive via Custom Method	T1027	Defense Evasion	Obfuscated Files or Information
		T1135	Discovery	Network Share Discovery	T1046	Discovery	Network Service Discovery
		T1021.001	Lateral Movement	Remote Desktop Protocol	T1572	Command and Control	Protocol Tunneling
		T1071.004	Command and Control	DNS	T1568	Command and Control	Dynamic Resolution
		T1036	Defense Evasion	Masquerading	T1027.003	Defense Evasion	Steganography

References

1. Huang, K.; Lian, Y.; Feng, D.; Zhang, H.; Liu, Y.; Ma, X. Cyber security threat intelligence sharing model based on blockchain. *J. Comput. Res. Dev* **2020**, *57*, 836–846.
2. Schlette, D.; Böhm, F.; Caselli, M.; Pernul, G. Measuring and visualizing cyber threat intelligence quality. *International Journal of Information Security* **2021**, *20*, 21–38.
3. Abu, M.S.; Selamat, S.R.; Ariffin, A.; Yusof, R. Cyber threat intelligence–issue and challenges. *Indonesian Journal of Electrical Engineering and Computer Science* **2018**, *10*, 371–379.
4. J, B.D. The Pyramid of Pain. https://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html.
5. Oosthoek, K.; Doerr, C. Cyber threat intelligence: A product without a process? *International Journal of Intelligence and CounterIntelligence* **2021**, *34*, 300–315.
6. Conti, M.; Dargahi, T.; Dehghantanha, A. *Cyber threat intelligence: challenges and opportunities*; Springer, 2018.
7. MITRE. ATT&CK. https://attack.mitre.org/.
8. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Advances in neural information processing systems* **2017**, *30*.
9. Zhong, L.; Wu, J.; Li, Q.; Peng, H.; Wu, X. A comprehensive survey on automatic knowledge graph construction. *arXiv preprint arXiv:2302.05019* **2023**.
10. Kim, H.; Kim, H.; others. Comparative experiment on TTP classification with class imbalance using oversampling from CTI dataset. *Security and Communication Networks* **2022**, *2022*.
11. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* **2002**, *16*, 321–357.

12. Wei, J.; Zou, K. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196* **2019**.
13. Rahman, M.R.; Williams, L. From Threat Reports to Continuous Threat Intelligence: A Comparison of Attack Technique Extraction Methods from Textual Artifacts. *arXiv preprint arXiv:2210.02601* **2022**.
14. Liu, J.; Yan, J.; Jiang, J.; He, Y.; Wang, X.; Jiang, Z.; Yang, P.; Li, N. TriCTI: an actionable cyber threat intelligence discovery system via trigger-enhanced neural network. *Cybersecurity* **2022**, *5*, 8.
15. Fang, L.; Lee, G.G.; Zhai, X. Using gpt-4 to augment unbalanced data for automatic scoring. *arXiv preprint arXiv:2310.18365* **2023**.
16. Fang, Y.; Li, X.; Thomas, S.W.; Zhu, X. Chatgpt as data augmentation for compositional generalization: A case study in open intent detection. *arXiv preprint arXiv:2308.13517* **2023**.
17. Dai, H.; Liu, Z.; Liao, W.; Huang, X.; Cao, Y.; Wu, Z.; Zhao, L.; Xu, S.; Liu, W.; Liu, N.; others. AugGPT: Leveraging ChatGPT for Text Data Augmentation. *arXiv preprint arXiv:2302.13007* **2023**.
18. Liu, C.; Wang, J.; Chen, X. Threat intelligence ATT&CK extraction based on the attention transformer hierarchical recurrent neural network. *Applied Soft Computing* **2022**, *122*, 108826.
19. Yu, Z.; Wang, J.; Tang, B.; Lu, L. Tactics and techniques classification in cyber threat intelligence. *The Computer Journal* **2023**, *66*, 1870–1881.
20. Yu, Z.; Wang, J.; Tang, B.; Ge, W. Research on the classification of cyber threat intelligence techniques and tactics based on attention mechanism and feature fusion. *Journal of Sichuan University (Natural Science Edition)* **2022**, *59*, 053003.
21. Ge, W.; Wang, J.; Tang, H.; Yu, Z.; Chen, B.; Yu, J. RENet: tactics and techniques classifications for cyber threat intelligence with relevance enhancement. *Journal of Sichuan University (Natural Science Edition)* **2022**, *59*, 023004.
22. Aghaei, E.; Al-Shaer, E. CVE-driven Attack Technique Prediction with Semantic Information Extraction and a Domain-specific Language Model. *arXiv preprint arXiv:2309.02785* **2023**.
23. Chae, Y.; Davidson, T. Large language models for text classification: From zero-shot learning to fine-tuning. *Open Science Foundation* **2023**.
24. Hegselmann, S.; Buendia, A.; Lang, H.; Agrawal, M.; Jiang, X.; Sontag, D. Tabllm: Few-shot classification of tabular data with large language models. International Conference on Artificial Intelligence and Statistics. PMLR, 2023, pp. 5549–5581.
25. Balkus, S.V.; Yan, D. Improving short text classification with augmented data using GPT-3. *Natural Language Engineering* **2022**, pp. 1–30.
26. Møller, A.G.; Dalsgaard, J.A.; Pera, A.; Aiello, L.M. Is a prompt and a few samples all you need? Using GPT-4 for data augmentation in low-resource classification tasks. *arXiv preprint arXiv:2304.13861* **2023**.
27. Puri, R.S.; Mishra, S.; Parmar, M.; Baral, C. How many data samples is an additional instruction worth? *arXiv preprint arXiv:2203.09161* **2022**.
28. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; others. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* **2023**.
29. Mangrulkar, S.; Gugger, S.; Debut, L.; Belkada, Y.; Paul, S.; Bossan, B. PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods. https://github.com/huggingface/peft, 2022.
30. Hu, E.J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W. LoRA: Low-Rank Adaptation of Large Language Models. International Conference on Learning Representations, 2022.
31. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; Du, Y.; Yang, C.; Chen, Y.; Chen, Z.; Jiang, J.; Ren, R.; Li, Y.; Tang, X.; Liu, Z.; Liu, P.; Nie, J.Y.; Wen, J.R. A Survey of Large Language Models. *arXiv preprint arXiv:2303.18223* **2023**.
32. Baichuan. Baichuan 2: Open Large-scale Language Models. *arXiv preprint arXiv:2309.10305* **2023**.
33. Team, B. BlueLM: An Open Multilingual 7B Language Model. https://github.com/vivo-ai-lab/BlueLM, 2023.
34. Zeng, A.; Liu, X.; Du, Z.; Wang, Z.; Lai, H.; Ding, M.; Yang, Z.; Xu, Y.; Zheng, W.; Xia, X.; others. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414* **2022**.
35. Du, Z.; Qian, Y.; Liu, X.; Ding, M.; Qiu, J.; Yang, Z.; Tang, J. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 320–335.

36. Muennighoff, N.; Wang, T.; Sutawika, L.; Roberts, A.; Biderman, S.; Scao, T.L.; Bari, M.S.; Shen, S.; Yong, Z.X.; Schoelkopf, H.; Tang, X.; Radev, D.; Aji, A.F.; Almubarak, K.; Albanie, S.; Alyafeai, Z.; Webson, A.; Raff, E.; Raffel, C. Crosslingual Generalization through Multitask Finetuning, 2023, [arXiv:cs.CL/2211.01786].
37. Bai, J.; Bai, S.; Chu, Y.; Cui, Z.; Dang, K.; Deng, X.; Fan, Y.; Ge, W.; Han, Y.; Huang, F.; Hui, B.; Ji, L.; Li, M.; Lin, J.; Lin, R.; Liu, D.; Liu, G.; Lu, C.; Lu, K.; Ma, J.; Men, R.; Ren, X.; Ren, X.; Tan, C.; Tan, S.; Tu, J.; Wang, P.; Wang, S.; Wang, W.; Wu, S.; Xu, B.; Xu, J.; Yang, A.; Yang, H.; Yang, J.; Yang, S.; Yao, Y.; Yu, B.; Yuan, H.; Yuan, Z.; Zhang, J.; Zhang, X.; Zhang, Y.; Zhang, Z.; Zhou, C.; Zhou, J.; Zhou, X.; Zhu, T. Qwen Technical Report. *arXiv preprint arXiv:2309.16609* **2023**.
38. Fayyazi, R.; Yang, S.J. On the Uses of Large Language Models to Interpret Ambiguous Cyberattack Descriptions. *arXiv preprint arXiv:2306.14062* **2023**.
39. Yin, S.; Fu, C.; Zhao, S.; Li, K.; Sun, X.; Xu, T.; Chen, E. A Survey on Multimodal Large Language Models. *arXiv preprint arXiv:2306.13549* **2023**.
40. Fu, C.; Chen, P.; Shen, Y.; Qin, Y.; Zhang, M.; Lin, X.; Yang, J.; Zheng, X.; Li, K.; Sun, X.; Wu, Y.; Ji, R. MME: A Comprehensive Evaluation Benchmark for Multimodal Large Language Models. *arXiv preprint arXiv:2306.13394* **2023**.
41. Yin, S.; Fu, C.; Zhao, S.; Xu, T.; Wang, H.; Sui, D.; Shen, Y.; Li, K.; Sun, X.; Chen, E. Woodpecker: Hallucination Correction for Multimodal Large Language Models. *arXiv preprint arXiv:2310.16045* **2023**.
42. Fu, C.; Zhang, R.; Wang, Z.; Huang, Y.; Zhang, Z.; Qiu, L.; Ye, G.; Shen, Y.; Zhang, M.; Chen, P.; Zhao, S.; Lin, S.; Jiang, D.; Yin, D.; Gao, P.; Li, K.; Li, H.; Sun, X. A Challenger to GPT-4V? Early Explorations of Gemini in Visual Expertise. *arXiv preprint arXiv:2312.12436* **2023**.