AD-LLM: Benchmarking Large Language Models for Anomaly Detection

Anonymous ACL submission

Abstract

Anomaly detection (AD) is an important machine learning task with many real-world uses, including fraud detection, medical diagnosis, and industrial monitoring. Within natural language processing (NLP), AD helps detect issues like spam, misinformation, and unusual user activity. Although large language models (LLMs) have had a strong impact on tasks such as text generation and summarization, their potential in AD has not been studied enough. This paper introduces AD-LLM, the first benchmark that evaluates how LLMs can help with NLP anomaly detection. We examine three key tasks: (*i*) zero-shot detection, using LLMs' pre-trained knowledge to perform AD without task-specific training; (*ii*) data augmentation, generating synthetic data and category descriptions to improve AD models; and (*iii*) model selection, using LLMs to suggest unsupervised AD models. Through experiments with different datasets, we find that LLMs can work well in zero-shot AD, that carefully designed augmentation methods are useful, and that explaining model selection for specific datasets remains challenging. Based on these results, we outline six future research directions on LLMs for AD.

1 Introduction

Anomaly detection (AD) is an important topic in machine learning (ML) that identifies samples differing from the general distribution (Zhao et al., 2019; Liu et al., 2024b). This ability is critical for many practical applications, such as fraud detection (Abdallah et al., 2016), medical diagnosis (Fernando et al., 2021), software engineering (Sun et al., 2022), and industrial system monitoring (Sun et al., 2023). Within natural language processing (NLP), AD is also important for finding unusual text instances, which is needed for detecting spam (Rao et al., 2021), misinformation (Islam et al., 2020), or unusual user behavior (Xue et al., 2023).

In the current era of large language models (LLMs), we ask how AD can make use of their capabilities and what the current level of integration looks like. While LLMs have brought large improvements to areas such as text generation, summarization, and translation, their possible benefits for AD, especially in NLP, have received some attention (Li et al., 2024a; Xu and Ding, 2024) but have not been studied in detail.

This work presents *the first comprehensive benchmark*, called AD-LLM, to study the roles and potential of LLMs in NLP anomaly detection. Our analysis focuses on three key tasks that are central in AD research and in practice (Figure 1):

- (*i*) **LLM for Anomaly Detection** (§3): Many AD tasks lack enough labeled data, making it hard to train models from scratch (Han et al., 2022). LLMs, with their pre-trained knowledge, can perform zero-shot AD (Xu and Ding, 2024).
- (*ii*) **LLM for Data Augmentation** (§4): AD tasks often suffer from unbalanced or limited data (Yoo et al., 2024; Li et al., 2023). For example, only a few insurance fraud samples may be available (Bauder and Khoshgoftaar, 2018). Generative LLMs may produce synthetic data to strengthen AD cost-effectively.
- (*iii*) **LLM for Model Selection** (§5): Picking a good AD model usually needs many trials and domain insights (Jiang et al., 2024a), and current choices in practice are often random (Zhao et al., 2021). LLMs, with the prior knowledge and ability to reason, may be able to suggest suitable AD models and save human effort.

Collectively, these three tasks tackle fundamental AD challenges from multiple angles: rapidly detecting anomalies with minimal supervision, enriching limited datasets for more robust learning, and guiding model selection without extensive domain expertise. As a result, AD-LLM not only improves individual AD components but also demonstrates *how LLMs can streamline the entire process*—from raw data to reliable, actionable insights.

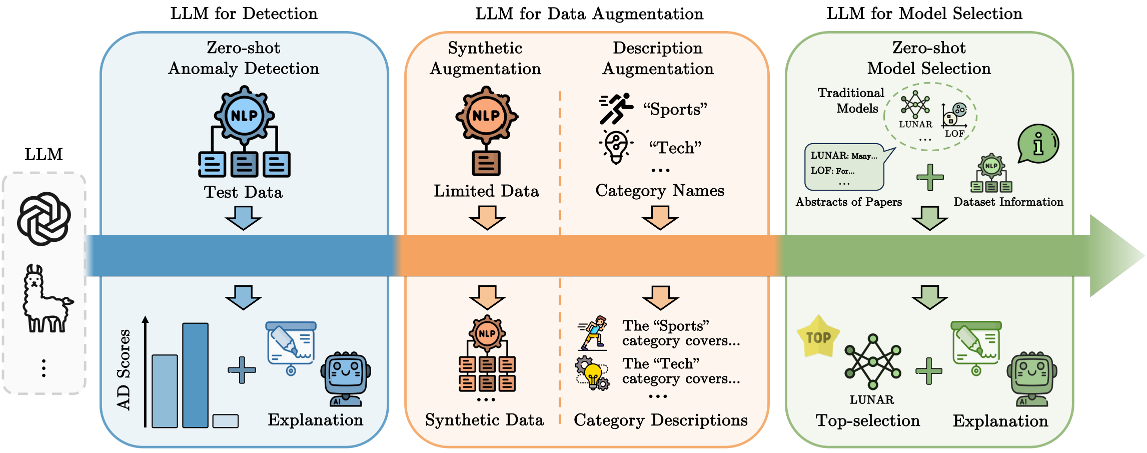

Figure 1: AD-LLM examines how LLMs contribute to three key AD tasks: (**Task 1**, §3) Zero-shot detection (left), where LLMs directly identify anomalies and provide explanations without task-specific training data; (**Task 2**, §4) Data augmentation (center), where LLMs generate synthetic samples and produce category descriptions to alleviate data scarcity and improve semantic reasoning; and (**Task 3**, §5) Model selection (right), where LLMs analyze dataset attributes and model descriptions to recommend suitable AD models along with justifications.

Key Takeaways. Our results reveal several noteworthy insights: (*i*) LLMs can achieve superior zero-shot AD performance, often outpacing conventional methods without relying on task-specific data; (*ii*) Enriching LLM inputs with additional context—such as anomaly category names or descriptive prompts—further boosts detection quality; (*iii*) Employing LLM-driven data augmentation enhances AD performance, though the effectiveness varies with model complexity and dataset properties; (*iv*) LLM-based model selection can approach top-performing baselines, but improving interpretability and providing dataset-specific rationales remains an open area. These suggest future work that systematically integrates external knowledge, refines prompt engineering, and develops strategies to ensure more transparent, context-sensitive LLM recommendations in AD tasks.

Contributions. This paper makes the following key contributions:

- **The First Comprehensive LLM-based AD benchmark.** We introduce AD-LLM, a unified evaluation framework that examines how LLMs address three core AD tasks—detection, data augmentation, and model selection.
- **Systematic and In-depth Experimental Analysis.** Through extensive experiments across multiple datasets, we show that LLMs can achieve strong zero-shot AD performance, boost AD methods by generating synthetic data or descriptive prompts, and recommend effective AD models w/o relying on historical performance data.
- **Reproducibility and Accessibility.** We release AD-LLM under the MIT License at `https://anonymous.4open.science/r/AD-LLM-F088`, providing a platform for the community to explore advanced applications of LLMs in AD.

2 Preliminaries on AD-LLM

2.1 Related Work

Recent studies have explored the role of LLMs in AD, highlighting both opportunities and challenges. (Xu and Ding, 2024) proposes a taxonomy categorizing LLMs as either detection or generative tools, but their work lacks experimental benchmarks. Similarly, (Jiang et al., 2024b) presents *MMAD*, a benchmark designed for industrial AD, focusing on image datasets yet limiting its applicability to other modalities. (Liu et al., 2024a) evaluates LLMs like Llama for out-of-distribution (OOD) detection, demonstrating the effectiveness of cosine distance detectors with isotropic embeddings. However, their work primarily focuses on traditional metrics and lacks insights into advanced LLM capabilities such as data augmentation and zero-shot detection.

Our work, *AD-LLM*, bridges these gaps by introducing a comprehensive benchmark for evaluating LLMs in anomaly detection across diverse tasks. This makes *AD-LLM* a significant step toward advancing LLM-driven anomaly detection.

2.2 Datasets and Traditional Baselines

Our experiments encompass five NLP AD datasets sourced from (Li et al., 2024c), derived from classification datasets. Each dataset contains text samples from multiple categories, with one designated as the anomaly category. The training data includes only normal samples. See the detailed information on datasets in Appx. A.1

We compare LLM-based AD with 18 traditional training-based unsupervised methods evaluated in (Li et al., 2024c) and leverage LLMs to enhance them. These baselines can be categorized into two groups: (1) End-to-end algorithms that directly process raw text data to produce AD results and (2) Two-step methods that first create text embeddings using language models and then apply traditional AD techniques to those embeddings. See a complete list of methods in Appx. A.2.

2.3 Common Experimental Settings

Evaluation Metrics. We evaluate the AD performance using two commonly used metrics (Han et al., 2022): (1) the Area Under the Receiver Operating Characteristic Curve (i.e., AUROC) and (2) the Area Under the Precision-Recall Curve (i.e., AUPRC). Both are the higher, the better.

LLMs and Hardware. We select two LLMs as main backbones: (1) Llama 3.1 8B Instruct (Dubey et al., 2024) and (2) GPT-4o (OpenAI, 2024a). For brevity, we refer to the "Llama 3.1 8B Instruct" as "Llama 3.1". Llama 3.1 runs on NVIDIA RTX 6000 Ada, 48 GB RAM workstations. GPT-4o is accessed through Azure OpenAI API with the "2024-08-01-preview" version and OpenAI official API. Seed is set = 42 for reproducibility. Specific experimental settings will be highlighted separately in each subsequent task.

3 Task 1: LLM for Zero-shot Detection

3.1 Motivation

Classical AD methods often require extensive training data—either labeled for supervised methods or unlabeled for unsupervised ones—which is time-consuming and costly (Han et al., 2022). In addition, setting up and tuning these models for real-world scenarios can be challenging and slow.

LLMs offer a practical alternative (Xu and Ding, 2024). With their broad pre-trained knowledge, they can perform zero-shot detection without additional training data. Their ability to understand language context and semantics makes them suitable for recognizing anomalies by logical reasoning. They can also explain their predictions, improving interpretability and trustworthiness (Huang et al., 2024b), which is important in sensitive domains such as healthcare, finance, and cybersecurity.

3.2 Problem Statement and Designs

Problem 1 (Zero-shot AD via LLMs) *Given a test set $\mathcal{D}_{test} = \{x_1, x_2, \ldots, x_n\}$ of text samples, where each sample x_i belongs to either a normal category or an anomaly category, the objective is to identify the anomalous samples using a pre-trained LLM f_{LLM} in a zero-shot setting without any task-specific training data.*

Evaluation Protocol. We focus on the ability of LLMs to detect anomalies without additional training data. We consider two settings, each reflecting different levels of prior knowledge:

- **Normal Only:** We provide only the normal category name(s) $\mathcal{C}_{\text{normal}}$. This matches scenarios where normal behavior is known but anomalies are uncertain or emerging.
- **Normal + Anomaly:** We provide both normal and anomaly category names, $\mathcal{C}_{\text{normal}}$ and $\mathcal{C}_{\text{anomaly}}$. This setting reflects situations where some information on anomalies is available, helping the LLM reason about what is anomalous.

The detection process is defined as:

$$\begin{aligned} \mathcal{P} &= T(x_i, \mathcal{C}_{\text{normal}}, \mathcal{C}^*_{\text{anomaly}}) \\ (r, s) &= f_{\text{LLM}}(\mathcal{P}) \end{aligned} \tag{1}$$

Here, $T(\cdot)$ constructs the prompt $\mathcal{P}$ for a test sample x_i, including known category information. The anomaly category $\mathcal{C}_{\text{anomaly}}$ is included only in the "Normal + Anomaly" setting, denoted as $\mathcal{C}^*_{\text{anomaly}}$. The LLM f_{LLM} processes the prompt to produce an anomaly score s and an explanation r that describes the reasoning. This setup allows a systematic evaluation of LLMs in zero-shot AD, using prompt-based inference to handle different levels of prior knowledge. See details in Appx. B.

3.3 Results, Insights, and Future Directions

We select Llama 3.1 and GPT-4o as zero-shot detectors with temperature = 0 for stable outputs.

LLMs are effective in zero-shot AD, surpassing existing training-based AD algorithms. We compare Llama 3.1-based and GPT-4o-based zero-shot detectors with baseline methods across five datasets in Table 1. GPT-4o consistently outperforms all baselines, and Llama 3.1 often ranks near the top. Despite operating with limited prior information, LLMs exhibit significant potential for anomaly detection tasks. These results highlight the strength of LLMs in zero-shot AD scenarios.

Additional context helps. Table 1 shows improved AUROC and AUPRC when using both normal and anomaly categories ("Normal + Anomaly") compared to using only normal categories ("Normal Only"). By providing more contextual information, LLMs better distinguish anomalous samples,

Table 1: Performance comparison of LLM-based detectors and baseline methods across five datasets, evaluated under two settings as described in §3.2 with AUROC and AUPRC as the metrics (higher (↑), the better). We list the best and the second-best baseline methods in each dataset. Complete results are provided in Appx. A2. The **best** results are highlighted in bold, and the second-best results are underlined. GPT-4o achieves the best performance consistently across all datasets, while Llama 3.1 also shows competitive performance, highlighting the effectiveness of LLM-based zero-shot anomaly detection.

Settings	**AG News**		**BBC News**		**IMDB Reviews**		**N24 News**		**SMS Spam**	
	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑
					Llama 3.1 8B Instruct					
(1) with $\mathcal{C}_{\text{normal}}$	0.8226	0.4036	0.7910	0.3602	0.7373	0.3474	0.6267	0.1130	0.7558	0.2884
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.8754	0.3998	0.8612	0.3960	0.8625	0.4606	0.8784	0.3802	0.9487	0.6361
					GPT-4o					
(1) with $\mathcal{C}_{\text{normal}}$	**0.9332**	**0.7207**	0.9574	0.8432	0.9349	0.7823	0.7674	0.3252	0.7940	0.5568
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.9293	0.6310	**0.9919**	**0.9088**	**0.9668**	**0.8465**	**0.9902**	**0.9009**	**0.9862**	**0.8953**
Best Baselines	**OpenAI + LUNAR**		**OpenAI + LUNAR**		**OpenAI + ECOD**		**OpenAI + LUNAR**		**DATE**	
	0.9226	0.6918	0.9732	0.8653	0.7366	0.5165	0.8320	0.4425	0.9398	0.6112
Second-best Baseline	**OpenAI + LOF**		**OpenAI + LOF**		**OpenAI + DeepSVDD**		**OpenAI + LOF**		**OpenAI + LOF**	
	0.8905	0.5443	0.9558	0.7714	0.6563	0.3278	0.7806	0.2248	0.7862	0.2450

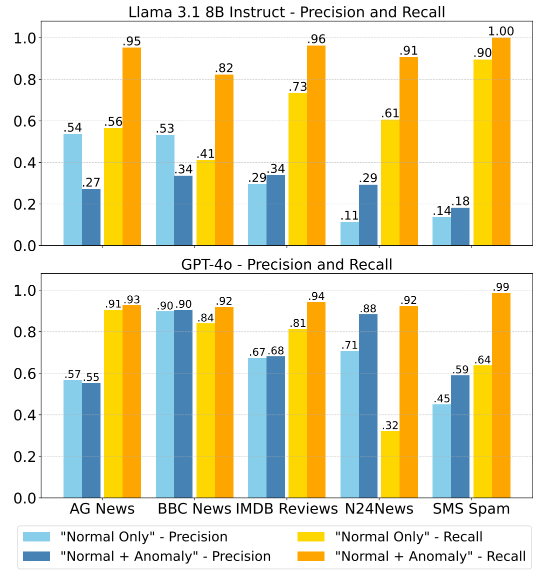

Figure 2: LLM-based AD precision-recall comparison between two settings across five datasets. "Normal Only" and "Normal + Anomaly" are two settings discussed in §3.2. The anomaly score threshold = 0.5 is used to decide the anomaly. Setting "Normal + Anomaly" outperforms setting "Normal Only" in most cases. In the "AG News" and "BBC News" datasets, there is a trade-off between precision and recall across two settings.

increasing detection effectiveness. These results suggest that augmenting LLMs with richer information enhances their ability to detect anomalies.

Trade-offs in detection focus. Figure 2 presents precision-recall comparisons for the two settings. Including anomaly information generally improves recall and sometimes precision. However, for "AG News" and "BBC News," we observe a recall gain at the cost of precision. This indicates that adding anomaly details may shift the LLM's detection balance, emphasizing one metric over another. Thus, one must carefully select the information provided to LLMs, as it significantly impacts their detection priorities and the balance among metrics.

Future Direction 1: Improve Context Integration. Providing additional context improves detection, as seen in "Normal + Anomaly." Future work may involve more systematic ways to integrate domain-specific details based on the detection priority (e.g., precision vs. recall), such as prompt design or retrieval-augmented methods (Gao et al., 2023).

Future Direction 2: Optimize for Real-world Deployment. Despite their effectiveness, LLM-based zero-shot AD is inherently time-consuming and costly (Sinha et al., 2024). Reducing computational overhead is important for deploying LLMs in real settings, especially for AD applications, which are often time-critical. Methods like quantization (Dettmers et al., 2023; Xiao et al., 2023), pruning (Sun et al., 2024; Fu et al., 2024), and knowledge distillation (Wang et al., 2024b; Fu et al., 2023) can help reduce the model size and inference time while maintaining good performance.

4 Task 2: LLM for Data Augmentation

4.1 Motivation

Data augmentation (DA) in AD aims to produce additional samples to improve model training under data scarcity (Yoo et al., 2023). However, traditional methods often struggle to capture the complexity of natural language, potentially causing a shift in domain characteristics (Feng et al., 2021). LLMs offer a solution, using their broad pre-trained knowledge and autoregressive learning objectives to generate contextually relevant data with better semantic understanding (Xu and Ding, 2024).

In addition, LLMs can generate textual descriptions (Xu and Ding, 2024) that assist the LLM-based detectors in §3. For example, by producing descriptions of **known categories**, LLMs help detectors establish distant associations between normal and anomalous samples (Menon and Vondrick, 2022; Zhu et al., 2024).

Thus, We examine two approaches that address data scarcity and improve semantic reasoning:

1. (§4.2) generates *synthetic samples* to improve training-based AD models.
2. (§4.3) produces *category descriptions* to refine prompts and enhance LLM-based detectors.

4.2 Generating Synthetic Samples for Training-based AD Models

Problem 2 (Synthetic DA via LLMs) *Given a small training set* $\mathcal{D}_{small\_train} = \{x_1, x_2, \ldots, x_m\}$ *of normal samples, the goal is to produce a synthetic dataset* $\mathcal{D}_{synth} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n\}$ *using a pre-trained LLM* f_{LLM}*. The combined dataset* $\mathcal{D}_{DA} = \mathcal{D}_{small\_train} \cup \mathcal{D}_{synth}$ *is used to train an unsupervised AD method* M*, improving performance compared to using* $\mathcal{D}_{small\_train}$ *alone.*

Evaluation Protocol. To evaluate the impact of LLM-generated synthetic data, we set unsupervised AD baselines listed in §A.2 in a scenario with limited training data. LLMs are then utilized to generate a synthetic training dataset. However, direct prompting often leads to highly repetitive outputs, even with high decoding temperatures (Long et al., 2024). Additionally, LLMs face constraints such as token limits (e.g., GPT-4's maximum output of 4,096 tokens) and challenges in processing long contexts (Gao et al., 2024). To address these issues, we adopt a multi-step strategy:

- *Step1: Keyword Generation:* Generate groups of keywords in one inquiry. Each group consists of three keywords w/ different levels of granularity: broad/general, intermediate, and fine-grained.
- *Step2: Sample Generation:* For each keyword group, generate one synthetic sample $\tilde{x}_i$.

By separating keyword generation from sample creation and enforcing different granularity levels, this approach introduces controlled variability and thematic breadth without causing the model to produce overly lengthy or repetitive data. The resulting synthetic samples are more likely to be contextually rich and semantically diverse. Further details are provided in Appx. C.1

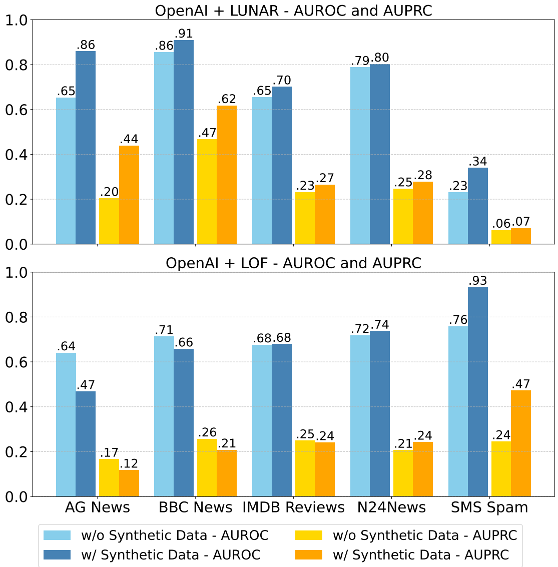

Figure 3: Performance comparison of AD baselines trained *with* and *without* LLM-generated synthetic data across five datasets. The use of LLM-generated synthetic data consistently enhances AD performance, with improvements varying by algorithm.

Results, Insights, and Future Directions. We apply the augmentation strategy to improve two unsupervised AD methods identified as strong baselines from Task 1 results: "OpenAI + LUNAR" (Goodge et al., 2022) and "OpenAI + LOF" (Breunig et al., 2000). We use GPT-4o with a temperature of 1.0 to introduce sufficient variability in synthetic samples.

LLM-generated synthetic data consistently improves AD performance. As shown in Figure 3, LLM-generated synthetic data provides overall performance improvements across anomaly detection methods. Notably, "OpenAI + LUNAR" demonstrates consistent gains across all five datasets, emphasizing the robustness of this approach. Significant improvements are observed in datasets like "SMS Spam" and "N24 News," where limited original training data posed significant challenges for traditional approaches. These results suggest that LLM-generated data is particularly effective in addressing data scarcity, complementing models like "LUNAR" that benefit from robust feature representations and semantic reasoning.

Improvement depends on model complexity and data characteristics. While "OpenAI + LUNAR" sees steady improvements, "OpenAI + LOF"—a simpler density-based model—exhibits variable outcomes. In data-scarce scenarios such as "SMS Spam" and "N24 News," synthetic data assists LOF. However, for datasets like "AG News" and "BBC News," performance declines, possibly due to the

Table 2: Performance (and △ changes) of LLM-based detectors **with augmented descriptions** under two settings in §3.2. The description generators and LLM-based detectors adopt the same LLM backbone. Values in brackets indicate changes compared to the results in Table 1. green denotes for improvements and red for declines. **Changes below** 0.03 **are not colored**, reflecting minor fluctuations. Augmented descriptions improve performance in most cases, particularly in AUROC, showcasing their effectiveness in enhancing LLM-based AD.

Settings	**AG News**		**BBC News**		**IMDB Reviews**		**N24 News**		**SMS Spam**	
	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑
					Llama 3.1 8B Instruct					
(1) with $\mathcal{C}_{\text{normal}}$	0.8081 (-0.0145)	0.3588 (-0.0448)	0.7802 (-0.0108)	0.3006 (-0.0596)	0.9039 (+0.1666)	0.6272 (+0.2798)	0.6651 (+0.0384)	0.1383 (+0.0253)	0.7456 (-0.0102)	0.2225 (-0.0659)
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.9046 (+0.0292)	0.5097 (+0.1099)	0.9089 (+0.0477)	0.6531 (+0.2571)	0.9351 (+0.0726)	0.6369 (+0.1763)	0.7900 (-0.0884)	0.2396 (-0.1406)	0.9413 (-0.0074)	0.7018 (+0.0657)
					GPT-4o					
(1) with $\mathcal{C}_{\text{normal}}$	0.9255 (-0.0077)	0.6985 (-0.0222)	0.9611 (+0.0037)	0.8162 (-0.0270)	0.9572 (+0.0223)	0.8307 (+0.0484)	0.8792 (+0.1118)	0.5399 (+0.2147)	0.8365 (+0.0425)	0.4765 (-0.0803)
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.9331 (+0.0038)	0.6659 (+0.0349)	0.9849 (-0.0070)	0.8998 (-0.0090)	0.9855 (+0.0187)	0.9219 (+0.0754)	0.9895 (-0.0007)	0.8680 (-0.0329)	0.9800 (-0.0062)	0.8889 (-0.0064)

synthetic samples shifting the underlying data distribution in ways that do not align with LOF's density assumptions. These results suggest that the effectiveness of LLM-based augmentation can depend on both the algorithm's complexity and the dataset's intrinsic structure.

Future Direction 3: Balance Diversity and Alignment in Synthetic Data. Future work should investigate techniques to balance the diversity of synthetic samples with their semantic alignment to real-world distributions. Excessive diversity risks producing samples that deviate too far from the target domain, while insufficient diversity may fail to address data scarcity and limit generalization (Guo and Chen, 2024). Potential strategies include adjusting the prompt engineering process, using retrieval-augmented LLMs, embedding-based filters to steer generation (O'Neill et al., 2023), and incorporating human-in-the-loop interventions (Chung et al., 2023) to refine synthetic data quality and improve downstream AD performance.

4.3 Generating Category Descriptions for LLM-based Detectors

Problem 3 (Description DA via LLMs) *Given category names $\mathcal{C}_{normal}$ and, optionally, $\mathcal{C}_{anomaly}$, the objective is to generate comprehensive textual descriptions d_{normal} and $d_{anomaly}$ using a pre-trained LLM f_{LLM}. These descriptions are then incorporated into the prompts of LLM-based detectors, aiming to improve their performance compared to using category names alone.*

Evaluation Protocol. Extending the zero-shot detection from §3, we employ LLMs to produce category descriptions that offer richer semantic signals beyond simple category names. Specifically, for each normal and anomaly category, we generate d_{normal} and d_{anomaly} based on the category names and the dataset's context. These descriptions can highlight distinctive features, typical lexical patterns, or behavioral characteristics that define normal or anomalous classes. By incorporating these descriptions into the prompt, we update Eq. (1) as:

$$\mathcal{P} = T\big(x_i, (\mathcal{C}_{\text{normal}}, \boxed{d_{\text{normal}}}), \\ (\mathcal{C}_{\text{anomaly}}, \boxed{d_{\text{anomaly}}})^*\big) \tag{2}$$

where $(\mathcal{C}_{\text{anomaly}}, d_{\text{anomaly}})^*$ applies only in the "Normal + Anomaly" setting (see §3.2). By enriching category names with category descriptions (highlighted with blue boxes), we enhance the LLM's ability to reason about subtle category distinctions and domain-relevant cues. More details are provided in Appx. C.2.

Results, Insights, and Future Directions. We utilize Llama 3.1 and GPT-4o to generate category descriptions. To balance the diversity and precision of the generation, we set the temperature to 0.5.

Augmented descriptions improve LLM-based AD. As shown in Table 2, incorporating category descriptions increases AUROC scores in most datasets. This suggests that the added semantic information helps LLM-based detectors discriminate anomalous samples more effectively, especially when the dataset's inherent structure aligns well with the contextual signals embedded in the descriptions. For example, in datasets like "IMDB Reviews" and "BBC News," providing richer textual representations of normal and anomalous classes translates to noticeable gains in both metrics.

Trade-offs arise between different metrics. While AUROC improvements are widespread, some datasets exhibit declines in AUPRC, indicating that enhanced semantic cues may not uniformly boost all aspects of detection. This trade-off could occur if broad, high-level descriptions cause certain anomalies to appear more similar to normal samples, reducing precision. For instance, in “N24 News” and “SMS Spam,” the provided descriptions might introduce overlapping attributes between categories, making it more challenging to maintain high precision. These results underscore the importance of calibrating the level and type of detail included in category descriptions to suit the specific dataset characteristics.

Future Direction 4: Select Representative Samples. An effective way to refine category descriptions is to ground them in representative samples from the dataset. Sampling strategies based on clustering (Axiotis et al., 2024) or diversity maximization (Moumoulidou et al., 2020) can identify prototype examples that guide LLMs toward producing more tailored and context-aware descriptions. By referencing these representative samples, future approaches may generate descriptions that better reflect subtle category distinctions, ultimately improving both ranking quality and the balance between precision and recall.

5 Task 3: LLM for AD Model Selection

5.1 Motivation

Unsupervised model selection (UMS) is critical for identifying the most suitable AD model by aligning its features with the attributes of a given dataset and the task’s requirements. Given the diverse range of AD models available and the absence of a universal solution, effective UMS is essential to ensure optimal performance. Traditional UMS methods often rely on historical performance data or domain-specific expertise; however, such data may be unavailable or irrelevant for novel or evolving datasets (Zhao et al., 2021; Zhao, 2024).

LLMs offer a promising zero-shot alternative by utilizing their extensive pre-trained knowledge to analyze datasets and recommend suitable models without relying on past performance metrics (Qin et al., 2024). They can streamline the model selection process, reducing manual overhead and domain knowledge requirements while also improving adaptability to novel data scenarios.

5.2 Problem Statement and Designs

Problem 4 (Zero-shot UMS via LLMs) *Given a dataset $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$ and a set of AD models $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$, the task is to identify a suitable model $M^* \in \mathcal{M}$ using a pre-trained LLM f_{LLM}, based solely on provided information about the dataset and the candidate models.*

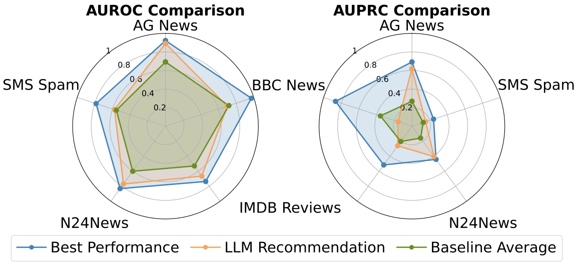

Figure 4: Comparison of AD performance across five datasets. “LLM Recommendation” denotes the average of models selected by querying the LLM five times (duplicates allowed). “Best Performance” indicates the highest performance among all AD models for each dataset, while “Baseline Average” shows the mean performance of all baseline AD models (as if choosing it randomly). Results show that models recommended by the LLM generally outperform the baseline average and, in some cases, approach the best-performing model, showing the potential of LLM-based UMS.

Evaluation Protocol. To enable LLM-based zero-shot UMS, we provide structured, detailed information of both the dataset and the candidate models:

- **Dataset Description:** dataset name, size, background, normal and anomaly categories, text-length statistics (average, maximum, minimum, and standard deviation), and representative samples of both normal and anomalous data. These attributes help the LLM understand the dataset’s structure, complexity, and potential challenges, and are generally easy to obtain for new datasets.
- **Model Description:** abstracts from published AD papers describing each candidate model. These abstracts highlight key model features, underlying assumptions, and targeted use cases. By examining these summaries, the LLM can align dataset attributes with model strengths, improving the relevance of its recommendations.

We then construct prompts that combine these datasets and model descriptions, asking the LLM to select and justify a recommended model. Further details about the prompt format and implementation can be found in Appx. D.

5.3 Results, Insights, and Future Directions

For the UMS scenario, which requires sophisticated reasoning, we use GPT-o1-preview (OpenAI, 2024c) with enhanced reasoning abilities.

LLM recommendations demonstrate strong potential. Figure 4 compares the detection performance across five datasets using three references: (*i*) the **best** achievable result from any model, representing the performance upper bound; (*ii*) the average performance of all baseline models, simulating a scenario where a user selects models at **random**; and (*iii*) the results from the LLM-recommended models (**ours**). The LLM-recommended models surpass the baseline average in most cases, and sometimes approach the best-performing model (e.g., on "IMDB Reviews" and "SMS Spam"). This indicates that LLMs can identify promising AD models using only public information, w/o reliance on historical performance or domain specialists.

LLM justifications lack dataset-specific insights. Although the LLM selects models that perform relatively well, its explanations often remain generic and do not clearly link model selection to specific dataset characteristics. For example, in the "AG News" dataset, the LLM alternated between recommending "OpenAI + LUNAR" and "OpenAI + ECOD," justifying choices with broad statements like "effective for high-dimensional data" or "parameter-free scalability." Such non-specific rationales diminish interpretability and user trust, especially when understanding the rationale behind model choice is important.

Future Direction 5: Refine Input Specificity and Addressing Biases. Future work should explore how to provide more dataset-specific details and mitigate potential LLM biases. Ambiguous or incomplete input information may cause the LLM to favor well-known models or those frequently encountered during training. Ensuring detailed and balanced inputs, and exploring how inherent biases in LLMs affect recommendations, will be important steps to improve the fairness and reliability of LLM-based UMS (Dai et al., 2024).

Future Direction 6: Enhancing Interpretability. Improving LLMs' capacity to produce transparent, dataset-tailored justifications for model selection decisions is key (Huang et al., 2024a). Techniques such as fine-tuning with richly annotated explanations or using prompt engineering to explicitly request structured reasoning can encourage the LLM to articulate clear, context-sensitive arguments.

6 Conclusion and Future Directions

In this work, we presented AD-LLM, the first comprehensive benchmark that integrates LLMs into three core aspects of anomaly detection in NLP: detection, data augmentation, and model selection. Our results show that LLMs hold strong potential across these tasks. For detection, LLMs demonstrate effective zero-shot and few-shot capabilities, often performing competitively against traditional methods without task-specific training. For data augmentation, both synthetic data generation and category description augmentation lead to improved performance for unsupervised and LLM-based AD methods. For model selection, LLMs can recommend suitable AD models using only general dataset and model descriptions. These findings highlight the emerging role of LLMs in making AD more flexible, data-efficient, and adaptable.

Future Directions. Investigating methods to systematically integrate external context into prompts can reduce variability and enhance precision. Techniques like quantization or hardware optimizations can lower the computational burden for real-world applications. Specialized augmentation approaches, designed to optimize user-prioritized metrics, may further improve AD performance. Finally, improving the specificity and clarity of LLM justifications can enhance interpretability and user trust, and expanding the AD-LLM benchmark to include additional tasks and applications in different fields (Huang et al., 2024b; Li et al., 2024b).

Broader Impact Statement

AD-LLM explores the application of LLMs in enhancing AD through zero-shot detection, data augmentation, and model selection. These contributions have the potential to significantly improve real-world AD systems in critical areas such as healthcare, finance, and cybersecurity. By enabling robust, adaptable, and efficient solutions for AD tasks, this research empowers practitioners to deploy systems that are responsive to novel challenges while reducing reliance on labeled data and extensive domain expertise.

Ethics Statement

This study adheres to ethical guidelines, emphasizing considerations around fairness, transparency, and privacy in developing and applying LLM-based AD systems. We emphasize the importance of evaluating and mitigating biases in LLM recommendations, ensuring that outputs are equitable and unbiased. Moreover, privacy is preserved by relying on public data and avoiding the collection of sensitive information. Also, note that we used ChatGPT exclusively to improve minor grammar in the final manuscript text.

Limitations

Despite promising results, several limitations remain. First, our evaluation is constrained to a narrow set of datasets with clear normal-anomaly distinctions, and our settings in AD and category descriptions in DA follow the structure of these datasets, limiting applicability to various domains with ambiguous anomaly definitions. Second, the synthetic data generation in DA is limited in scale, producing few samples per category, and challenges in scaling up remain unresolved. Third, UMS depends on simplistic input data and matching mechanisms. Furthermore, biases in LLM recommendations, such as favoring well-documented or familiar models, need further investigation. Finally, our study evaluates only a subset of popular LLMs, lacking a comprehensive assessment. Additionally, we do not explore few-shot learning or fine-tuning, which are widely adopted techniques for enhancing LLM performance and could offer valuable complementary insights for AD tasks.

References

Aisha Abdallah, Mohd Aizaini Maarof, and Anazida Zainal. 2016. Fraud detection system: A survey. *Journal of Network and Computer Applications*, 68:90–113.

Charu C Aggarwal. 2015. Outlier analysis. In *Data mining*, pages 75–79. Springer.

Kyriakos Axiotis, Vincent Cohen-Addad, Monika Henzinger, Sammy Jerome, Vahab Mirrokni, David Saulpic, David Woodruff, and Michael Wunder. 2024. Data-efficient learning via clustering-based sensitivity sampling: Foundation models and beyond. In *Forty-first International Conference on Machine Learning*.

Richard A Bauder and Taghi M Khoshgoftaar. 2018. The effects of varying class distribution on learner behavior for medicare fraud detection with imbalanced big data. *Health information science and systems*, 6:1–14.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104.

Yunkang Cao, Xiaohao Xu, Chen Sun, Xiaonan Huang, and Weiming Shen. 2023. Towards generic anomaly detection and understanding: Large-scale visual-linguistic model (gpt-4v) takes the lead. *arXiv preprint arXiv:2311.02782*.

John Chung, Ece Kamar, and Saleema Amershi. 2023. Increasing diversity while maintaining accuracy: Text data generation with large language models and human interventions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 575–593.

Sunhao Dai, Chen Xu, Shicheng Xu, Liang Pang, Zhenhua Dong, and Jun Xu. 2024. Bias and unfairness in information retrieval systems: New challenges in the llm era. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6437–6447.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: efficient finetuning of quantized llms (2023). *arXiv preprint arXiv:2305.14314*, 52:3982–3992.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. 2021. Deep learning for medical anomaly detection–a survey. *ACM Computing Surveys (CSUR)*, 54(7):1–37.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pages 10421–10430. PMLR.

Yonggan Fu, Zhongzhi Yu, Junwei Li, Jiayi Qian, Yongan Zhang, Xiangchi Yuan, Dachuan Shi, Roman Yakunin, and Yingyan Celine Lin. 2024. Amoeballm: Constructing any-shape large language models for efficient and instant deployment. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Muhan Gao, TaiMing Lu, Kuai Yu, Adam Byerly, and Daniel Khashabi. 2024. Insights into llm long-context failures: When transformers know but don't tell. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7611–7625.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Adam Goodge, Bryan Hooi, See-Kiong Ng, and Wee Siong Ng. 2022. Lunar: Unifying local outlier detection methods via graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6737–6745.

Xu Guo and Yiqiang Chen. 2024. Generative ai for synthetic data generation: Methods, challenges and the future. *arXiv preprint arXiv:2403.04190*.

Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. 2022. Adbench: Anomaly detection benchmark. *Advances in Neural Information Processing Systems*, 35:32142–32159.

Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, Hanchi Sun, Zhengliang Liu, Yixin Liu, Yijue Wang, Zhikun Zhang, Bertie Vidgen, Bhavya Kailkhura, Caiming Xiong, Chaowei Xiao, Chunyuan Li, Eric P. Xing, Furong Huang, Hao Liu, Heng Ji, Hongyi Wang, Huan Zhang, Huaxiu Yao, Manolis Kellis, Marinka Zitnik, Meng Jiang, Mohit Bansal, James Zou, Jian Pei, Jian Liu, Jianfeng Gao, Jiawei Han, Jieyu Zhao, Jiliang Tang, Jindong Wang, Joaquin Vanschoren, John Mitchell, Kai Shu, Kaidi Xu, Kai-Wei Chang, Lifang He, Lifu Huang, Michael Backes, Neil Zhenqiang Gong, Philip S. Yu, Pin-Yu Chen, Quanquan Gu, Ran Xu, Rex Ying, Shuiwang Ji, Suman Jana, Tianlong Chen, Tianming Liu, Tianyi Zhou, William Yang Wang, Xiang Li, Xiangliang Zhang, Xiao Wang, Xing Xie, Xun Chen, Xuyu Wang, Yan Liu, Yanfang Ye, Yinzhi Cao, Yong Chen, and Yue Zhao. 2024a. Position: TrustLLM: Trustworthiness in large language models. In *Forty-first International Conference on Machine Learning*.

Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, et al. 2024b. Position: Trustllm: Trustworthiness in large language models. In *International Conference on Machine Learning*, pages 20166–20270. PMLR.

Md Rafiqul Islam, Shaowu Liu, Xianzhi Wang, and Guandong Xu. 2020. Deep learning for misinformation detection on online social networks: a survey and new perspectives. *Social Network Analysis and Mining*, 10(1):82.

Minqi Jiang, Chaochuan Hou, Ao Zheng, Songqiao Han, Hailiang Huang, Qingsong Wen, Xiyang Hu, and Yue Zhao. 2024a. Adgym: Design choices for deep anomaly detection. *Advances in Neural Information Processing Systems*, 36.

Xi Jiang, Jian Li, Hanqiu Deng, Yong Liu, Bin-Bin Gao, Yifeng Zhou, Jialin Li, Chengjie Wang, and Feng Zheng. 2024b. Mmad: The first-ever comprehensive benchmark for multimodal large language models in industrial anomaly detection. *arXiv preprint arXiv:2410.09453*.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Jun Li, Cosmin I Bercea, Philip Müller, Lina Felsner, Suhwan Kim, Daniel Rueckert, Benedikt Wiestler, and Julia A Schnabel. 2024a. Language models meet anomaly detection for better interpretability and generalizability. *arXiv preprint arXiv:2404.07622*.

Li Li, Chenwei Wang, You Qin, Wei Ji, and Renjie Liang. 2023. Biased-predicate annotation identification via unbiased visual predicate representation. In *Proceedings of the 31st ACM International Conference on Multimedia*, page 4410–4420.

Lincan Li, Jiaqi Li, Catherine Chen, Fred Gui, Hongjia Yang, Chenxiao Yu, Zhengguang Wang, Jianing Cai, Junlong Aaron Zhou, Bolin Shen, et al. 2024b. Political-llm: Large language models in political science. *arXiv preprint arXiv:2412.06864*.

Yuangang Li, Jiaqi Li, Zhuo Xiao, Tiankai Yang, Yi Nian, Xiyang Hu, and Yue Zhao. 2024c. Nlp-adbench: Nlp anomaly detection benchmark. *arXiv preprint arXiv:2412.04784*.

Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George H Chen. 2022. Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12181–12193.

Bo Liu, Li-Ming Zhan, Zexin Lu, Yujie Feng, Lei Xue, and Xiao-Ming Wu. 2024a. How good are LLMs at out-of-distribution detection? In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8211–8222, Torino, Italia. ELRA and ICCL.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE.

Kay Liu, Yingtong Dou, Xueying Ding, Xiyang Hu, Ruitong Zhang, Hao Peng, Lichao Sun, and S Yu Philip. 2024b. Pygod: A python library for graph outlier detection. *Journal of Machine Learning Research*, 25(141):1–9.

Yezheng Liu, Zhe Li, Chong Zhou, Yuanchun Jiang, Jianshan Sun, Meng Wang, and Xiangnan He. 2019. Generative adversarial active learning for unsupervised outlier detection. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1517–1528.

Yilun Liu, Shimin Tao, Weibin Meng, Jingyu Wang, Wenbing Ma, Yuhang Chen, Yanqing Zhao, Hao Yang, and Yanfei Jiang. 2024c. Interpretable online log analysis using large language models with prompt strategies. In *Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension*, pages 35–46.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 11065–11082.

Andrei Manolache, Florin Brad, and Elena Burceanu. 2021. Date: Detecting anomalies in text via self-supervision of transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 267–277.

Sachit Menon and Carl Vondrick. 2022. Visual classification via description from large language models. *arXiv preprint arXiv:2210.07183*.

Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, et al. 2023. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*.

Zafeiria Moumoulidou, Andrew McGregor, and Alexandra Meliou. 2020. Diverse data selection under fairness constraints. *arXiv preprint arXiv:2010.09141*.

Charles O'Neill, Yuan-Sen Ting, Ioana Ciuca, Jack W Miller, and Thang Bui. 2023. Steering language generation: Harnessing contrastive expert guidance and negative prompting for coherent and diverse synthetic data generation. *CoRR*.

OpenAI. 2024a. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

OpenAI. 2024b. New embedding models and api updates.

OpenAI. 2024c. Openai o1 system card.

Yuehan Qin, Yichi Zhang, Yi Nian, Xueying Ding, and Yue Zhao. 2024. Metaood: Automatic selection of ood detection models. *arXiv preprint arXiv:2410.03074*.

Sanjeev Rao, Anil Kumar Verma, and Tarunpreet Bhatia. 2021. A review on social spam detection: Challenges, open issues, and future directions. *Expert Systems with Applications*, 186:115742.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402. PMLR.

Lukas Ruff, Yury Zemlyanskiy, Robert Vandermeulen, Thomas Schnake, and Marius Kloft. 2019. Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4061–4071.

Rohan Sinha, Amine Elhafsi, Christopher Agia, Matthew Foutter, Ed Schmerling, and Marco Pavone. 2024. Real-time anomaly detection and reactive planning with large language models. In *Robotics: Science and Systems*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Yongqian Sun, Daguo Cheng, Tiankai Yang, Yuhe Ji, Shenglin Zhang, Man Zhu, Xiao Xiong, Qiliang Fan, Minghan Liang, Dan Pei, et al. 2023. Efficient and robust kpi outlier detection for large-scale datacenters. *IEEE Transactions on Computers*, 72(10):2858–2871.

Zhensu Sun, Li Li, Yan Liu, Xiaoning Du, and Li Li. 2022. On the importance of building high-quality training datasets for neural code search. In *Proceedings of the 44th International Conference on Software Engineering*, page 1609–1620.

Jun Wang, Meng Fang, Ziyu Wan, Muning Wen, Jiachen Zhu, Anjie Liu, Ziqin Gong, Yan Song, Lei Chen, Lionel M Ni, et al. 2024a. Openr: An open source framework for advanced reasoning with large language models. *arXiv preprint arXiv:2410.09671*.

Xinfeng Wang, Jin Cui, Yoshimi Suzuki, and Fumiyo Fukumoto. 2024b. RDRec: Rationale distillation for LLM-based recommendation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 65–74, Bangkok, Thailand. Association for Computational Linguistics.

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

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

Ruiyao Xu and Kaize Ding. 2024. Large language models for anomaly and out-of-distribution detection: A survey. *arXiv preprint arXiv:2409.01980*.

Zhebin Xue, Qing Li, and Xianyi Zeng. 2023. Social media user behavior analysis applied to the fashion and apparel industry in the big data era. *Journal of Retailing and Consumer Services*, 72:103299.

Jaemin Yoo, Tiancheng Zhao, and Leman Akoglu. 2024. Data augmentation is a hyperparameter: Cherry-picked self-supervision for unsupervised anomaly detection is creating the illusion of success. *Transactions on Machine Learning Research*.

Jaemin Yoo, Yue Zhao, Lingxiao Zhao, and Leman Akoglu. 2023. Dsv: An alignment validation loss for self-supervised outlier model selection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 254–269. Springer.

Yue Zhao. 2024. Towards reproducible, automated, and scalable anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 22687–22687.

Yue Zhao, Zain Nasrullah, and Zheng Li. 2019. Pyod: A python toolbox for scalable outlier detection. *Journal of machine learning research*, 20(96):1–7.

Yue Zhao, Ryan Rossi, and Leman Akoglu. 2021. Automatic unsupervised outlier model selection. *Advances in Neural Information Processing Systems*, 34:4489–4502.

Jiaqi Zhu, Shaofeng Cai, Fang Deng, Beng Chin Ooi, and Junran Wu. 2024. Do llms understand visual anomalies? uncovering llm's capabilities in zero-shot anomaly detection. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 48–57.

Supplementary Material for AD-LLM

A Additional Details for Preliminaries

A.1 Datasets Details

As briefly discussed in §2.2, we select five NLD AD datasets with high quality and a proper size sourced from (Li et al., 2024c): AG News, BBC News, IMDB Reviews, N24News, SMS Spam. These datasets are originally intended for NLP classification tasks and contain text samples categorized into multiple groups, with one designated anomalous. The training data comprises only normal samples. Table A.1 provides a summary of dataset attributes, and Table A1 presents the statistics of datasets that will be utilized in our tasks.

Dataset	Avg.	Max.	Min.	Std.
AG News	190.1	959	35	61.7
BBC News	2,293.5	25,367	685	1,506.4
IMDB Reviews	1,289.2	12,498	65	980.5
N24 News	4,633.3	28,616	4	3,069.5
SMS Spam	78.7	790	4	60.8

Table A1: Statistics of datasets including average, maximum, minimum, standard deviation of text length.

A.2 Traditional Baselines Details

This study utilizes 18 traditional methods as baselines. We compare the performance of LLM-based anomaly detection methods with these baselines in §3 and further enhance the baselines with LLM-generated synthetic data, demonstrating the effectiveness of augmentation in §4.2.

These methods are categorized into two groups:

- **End-to-end Methods**. These methods directly process raw text data to generate AD results:
  - **CVDD**: Context Vector Data Description (Ruff et al., 2019). CVDD uses embeddings and self-attention to learn context vectors, detecting anomalies via deviations.
  - **DATE**: Detecting Anomalies in Text via Self-Supervision of Transformers (Manolache et al., 2021). DATE trains self-supervised transformers to identify anomalies in text.
- **Two-Step Methods**. These approaches first generate text embeddings using BERT (Kenton and Toutanova, 2019) or OpenAI's *text-embedding-3-large* (OpenAI, 2024b) and then apply traditional AD techniques to the embeddings.
  - **AE**: AutoEncoder (Aggarwal, 2015). AE uses high reconstruction errors to detect anomalies.
  - **DeepSVDD**: Deep Support Vector Data Description (Ruff et al., 2018). DeepSVDD identifies anomalies outside a hypersphere that encloses normal data representations.
  - **ECOD**: Empirical-Cumulative-distribution-based Outlier Detection (Li et al., 2022). ECOD flags point in distribution tails using empirical cumulative distributions.
  - **IForest**: Isolation Forest (Liu et al., 2008). IForest isolates anomalies with fewer splits in random feature-based partitions.
  - **LOF**: Local Outlier Factor (Breunig et al., 2000). LOF detects anomalies by comparing the local density of a point to its neighbors.
  - **SO_GAAL**: Single-Objective Generative Adversarial Active Learning (Liu et al., 2019). SO_GAAL generates adversarial samples to uncover anomalies in unsupervised settings.
  - **LUNAR**: Unifying Local Outlier Detection Methods via Graph Neural Networks (Goodge et al., 2022). LUNAR unifies and improves local outlier detection via graph neural networks.
  - **VAE**: Variational AutoEncoder (Kingma and Welling, 2014). VAE uses reconstruction probabilities to detect anomalies.

B Additional Details for Task 1

B.1 Prompt Details

Prompt design is crucial for zero-shot LLM-based detection, as the performance heavily relies on its instructiveness and clarity. As discussed in §3.2 about LLM-based zero-shot AD, we evaluate two settings based on varying levels of prior knowledge in the real world: "Normal Only" and "Normal + Anomaly." The LLM prompt template for setting "Normal Only" is provided in Table A9, and the prompt template for setting "Normal + Anomaly" is presented in Table A10. The prompt templates of the two settings are different in the ***definition of anomaly***, marked in red in Table A10.

We utilize a series of prompt engineering techniques, including:

- *Task Information* (Cao et al., 2023). It is essential to provide clear task information. We carefully define the detection scenario, the anomaly definition, and the rules to reduce hallucinations.
- *Chain-of-Thought (CoT)* (Wei et al., 2022). CoT prompting encourages LLMs to decompose their reasoning into sequential intermediate steps and organize information logically. We explicitly provide a completed chain of thoughts in the prompt.

Table A2: Detailed information of five datasets used in AD-LLM, including the original task, normal category(ies), anomaly category, the size of the training set, the size, and the anomaly ratio of the test set.

Dataset	Original Task	Normal Category(ies)	Anomaly Category	# Train	# Test	% Anomaly
AG News	AG news topics classification	Sports, Business, Sci/Tech	World	66,098	32,109	11.77%
BBC News	BBC news topics classification	Business, Politics, Sport, Tech	Entertainment	1,206	579	10.71%
IMDB Reviews	binary sentiment classification of IMDb movie reviews	Positive	Negative	17,417	8,952	16.61%
N24 News	New York Times news classification	Television, Your Money, Automobiles, Science, Economy, Dance, Travel, Technology, Sports, Movies, Music, Real Estate, Books, Education, Art & Design, Theater, Media, Style, Global Business, Well, Health, Fashion & Style, Opinion	Food	40,569	19,227	9.51%
SMS Spam	mobile phone SMS spam messages detection	Non-spam (Ham)	Spam	3,162	1,510	10.20%

- *Explanation and Implicit CoT*. We require an explanation r generated before the anomaly score s for each inquiry as shown in Eq. (1). When generating the explanation, LLMs **implicitly** create the CoT in the background (Liu et al., 2024c). This approach aligns with the auto-regressive nature of decoder-only LLMs, encouraging them to think carefully and logically before determining the anomaly score, thereby enhancing reliability.

In our experiments, we discovered that Llama 3.1 requires implicit CoT. Presenting the anomaly score s before the explanation r causes the Llama 3.1-based detector to crash and consistently outputs $s = 0$. This issue does not impact GPT-4o-based detectors. We attribute this to GPT-4o's significantly larger parameter count (8B), which grants it a stronger resilience to prompt changes.

B.2 Complete Baseline Results

In addition to the top two baseline results in §3.3, we provide the complete results for all 18 baseline methods in Table A8. We observe that Llama 3.1 outperforms most of these baselines, further supporting the efficacy of zero-shot AD via LLMs shown in Table 1. The baseline results presented in this study are derived from (Li et al., 2024c).

We present a bar chart in Figure 2 comparing the precision and recall LLM-based AD between two settings across five datasets. The detailed results, including precise numerical values, are provided in Table A3 for reference.

B.3 Error Analysis

In our experiments, LLMs occasionally encountered errors. Llama 3.1 sometimes entered infinite loops, repeatedly generating the same sentence until reaching the token limit, as illustrated in the AG News example shown in Text Box B.3, where the repetitive nonsensical response is highlighted in *italics*. However, the underlying causes of such behavior remain underexplored. For GPT-4o, outputs were sometimes restricted by OpenAI's safety filters when addressing potentially harmful content, such as politically sensitive topics.

We exclude these errors from our results but provide the error counts in both "Normal Only" and "Normal + Anomaly" settings for reference in Table A4 for Llama 3.1 and Table A5 for GPT-4o. Notably, the error counts vary between the two settings, suggesting that the triggers for errors, such as Llama's infinite loop or GPT-4o's safety filters, are sensitive to prompt variations. This occurs even though the prompts in both settings have similar semantic meanings.

An infinite loop example in Llama 3.1

Text sample: "This week's TravelWatch column profiles Anangu Tours, an Aborigine owned tour company in Australia's Red Center."
Response: {"reason": "The text sample is about a tour company in Australia, which relates to travel and geography, making it most closely align with the Sports category is not the best fit, but it does not fit into the Business or Sci/Tech categories, but it does fit into the Sports category is not the best fit, but it does fit into the category of travel which is related to Sports, but more closely related to the category of travel which is not listed, *but the closest is Sports, but it is more closely related to the category of travel which is not listed, but the closest is Sports, but it is more closely related to the category of travel which is not listed, but the closest is Sports, but it is more closely related to the category of travel which is not listed, but the closest is Sports...*

Table A3: LLM-based AD precision-recall comparison between two settings across five datasets. The anomaly score threshold $= 0.5$ is used to decide the anomaly. The better results for each detector are highlighted in bold.

Settings	**AG News**		**BBC News**		**IMDB Reviews**		**N24News**		**SMS Spam**	
	Precision ↑	Recall ↑	Precision ↑	Recall ↑	Precision ↑	Recall ↑	Precision ↑	Recall ↑	Precision ↑	Recall ↑
					Llama 3.1 8B Instruct					
(1) with $\mathcal{C}_{\text{normal}}$	**0.5364**	0.5645	**0.5319**	0.4098	0.2946	0.7328	0.1130	0.6053	0.1352	0.8961
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.2719	**0.9518**	0.3355	**0.8226**	**0.3379**	**0.9622**	**0.2930**	**0.9067**	**0.1816**	**1.000**
					GPT-4o					
(1) with $\mathcal{C}_{\text{normal}}$	**0.5673**	0.9058	0.8966	0.8387	0.6739	0.8130	0.7077	0.3206	0.4495	0.6364
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.5527	**0.9259**	**0.9048**	**0.9194**	**0.6803**	**0.9428**	**0.8828**	**0.9234**	**0.5891**	**0.9870**

Dataset	**“Normal Only”**	**“Normal + Anomaly”**
AG News	552	48
BBC News	0	3
IMDB Reviews	21	29
N24 News	299	898
SMS Spam	0	2

Table A4: Error count in Llama 3.1

Dataset	**“Normal Only”**	**“Normal + Anomaly”**
AG News	1	0
BBC News	0	0
IMDB Reviews	1	9
N24 News	0	0
SMS Spam	0	0

Table A5: Error count in GPT-4o

C Additional Details for Task 2

C.1 Generating Synthetic Samples Details

C.1.1 Evaluation Protocol and Prompt Details

As discussed in §4.2, we set a scenario with limited training data $\mathcal{D}_{\text{small\_train}} = \{x_1, x_2, \ldots, x_m\}$. Specifically, $\mathcal{D}_{\text{small\_train}}$ contains v samples for each normal category $\mathcal{C}^j_{\text{normal}} \in \mathcal{C}_{\text{normal}} = \{\mathcal{C}^1_{\text{normal}}, \ldots, \mathcal{C}^k_{\text{normal}}\}$, where k is the number of normal categories.

We employ a multi-step strategy in this task to mitigate repetitive outputs, token limit constraints, and difficulties in handling long contexts. The detailed pipeline is outlined below:

1. *Keywords Generation.* To ensure a consistent synthetic data distribution compared with the original training data, t groups of keywords are generated for each normal category $\mathcal{C}^j_{\text{normal}}$. We construct the prompt $\mathcal{P}_{\text{keywords}}$ using a template $T_{\text{keywords}}(\cdot)$ as shown in Table A11. This template utilizes `{name}` and `{original_task}` information from Table A.1. The prompt $\mathcal{P}_{\text{keywords}}$ is processed by the LLM $f_{\text{LLM}}(\cdot)$ to produce $t \times k$ groups of keywords $\mathcal{K} = \{\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_{t\times k}\}$. Each keyword group $\mathcal{K}_i$ contains three keywords with increasing levels of granularity from coarse to fine.
2. *Synthetic Sample Generation.* We iterate the groups of keywords, constructing $\mathcal{P}_{\text{synth}} = \{\mathcal{P}^1_{\text{synth}}, \mathcal{P}^2_{\text{synth}}, \ldots, \mathcal{P}^{t\times k}_{\text{synth}}\}$ using a template $T_{\text{synth}}(\cdot)$ as displayed in Table A12. Each prompt $\mathcal{P}^j_{\text{synth}}$ is fed into the LLM $f_{\text{LLM}}(\cdot)$ to generate a corresponding synthetic sample $\hat{x}_j$. Finally, we obtain a synthetic dataset $\mathcal{D}_{\text{synth}} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_{t\times k}\}$.

The pipeline is formally summarized as follows:

$$\begin{aligned}
\mathcal{P}_{\text{keywords}} &= T_{\text{keywords}}(\texttt{\{name\}}, \\
&\qquad\qquad \texttt{\{original\_task\}}) \\
\mathcal{K} &= f_{\text{LLM}}(\mathcal{P}_{\text{keywords}}) = \{\mathcal{K}_1, \ldots, \mathcal{K}_{t\times k}\} \\
\mathcal{P}_{\text{synth}} &= \{T_{\text{synth}}(\mathcal{K}_1), \ldots, T_{\text{synth}}(\mathcal{K}_{t\times k})\} \\
\mathcal{D}_{\text{synth}} &= \{f_{\text{LLM}}(\mathcal{P}^1_{\text{synth}}), \ldots, f_{\text{LLM}}(\mathcal{P}^{t\times k}_{\text{synth}})\}
\end{aligned} \tag{A1}$$

The prompt templates T_{keywords} and T_{synth} leverage the prompt techniques, including task information and CoT, as discussed in §B.1.

C.1.2 Experiments Details and Challenges

We set the number of samples from each normal category $\mathcal{C}^j_{\text{normal}}$ in the limited training set $\mathcal{D}_{\text{small\_train}}$ to $v = 10$. Similarly, the number of synthetic samples generated for each normal category $\mathcal{C}^j_{\text{normal}}$ in the synthetic set $\mathcal{D}_{\text{synth}}$ is set to $t = 50$ for the “AG New”, “BBC News”, “IMDB Reviews”, and “SMS Spam” datasets. For the “N24 News” dataset, we set $v = 3$ and $t = 12$ to account for its numerous normal categories.

In our experiments, GPT-4o was used for synthetic data generation. We observed that increasing t occasionally caused GPT-4o to terminate the keyword generation process before reaching the token limit. A similar issue occurred with Llama 3.1, even for smaller values of t. As a result, Llama 3.1 was excluded from this task. We presume these issues stem from the inherent challenges LLMs face in processing long contexts.

Table A6: Comparison of AD performance across five datasets. "LLM Recommendation" denotes the average of models selected by querying the LLM five times (duplicates allowed). "Best Performance" indicates the highest performance among all AD models for each dataset, while "Baseline Average" shows the mean performance of all baseline AD models (as if choosing it randomly).

Training Set	AG News		BBC News		IMDB Reviews		N24 News		SMS Spam	
	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑
LLM Recommendation	0.8908	0.6193	0.6992	0.1573	0.6652	0.2608	0.7706	0.4047	0.5774	0.1548
Baseline Average	0.6924	0.2685	0.7178	0.3574	0.5298	0.2038	0.6004	0.1585	0.5565	0.1277
Best Performance	0.9226	0.6918	0.9732	0.8653	0.7366	0.5165	0.8320	0.4425	0.7862	0.2450

A straightforward solution is generating keywords in multiple rounds with different temperatures and prompt templates. Further techniques, such as sample-wise decomposition and dataset-wise decomposition (Long et al., 2024), can be explored to optimize the synthetic data generation.

C.1.3 Complete Results

We present a bar chart in Figure 3 comparing the performance of AD baselines trained *with* and *without* LLM-generated synthetic data only on "OpenAI + LUNAR" and "OpenAI + LOF". The detailed results for all two-step methods using OpenAI, including precise numerical values, are provided in Table A7 for reference. These additional experiments on baselines follow the settings used in (Li et al., 2024c), except that the "batch_size" is set $= 4$ due to the amount of $\mathcal{D}_{\text{small\_train}}$ in AE, VAE, and DeepSVVD.

We observe the same trend discussed in §4.2: LLM-generated synthetic data improves AD performance in most cases, though the degree of enhancement varies based on model complexity and data characteristics.

C.1.4 Edges over LLM-based Zero-shot AD

At first glance, LLM-based zero-shot AD could eliminate the need to generate synthetic datasets for traditional models. However, they address different needs and offer complementary advantages. LLM-based zero-shot detection requires no task-specific training, offering easy deployment, adaptability across scenarios, and real-time inference—ideal for dynamic environments. However, its high computational cost can limit scalability for long-term or large-scale use.

In contrast, LLM-generated synthetic data enables the training of traditional models, significantly reducing inference costs for long-term or high-frequency detection tasks. Moreover, synthetic data can be a valuable resource for fine-tuning LLMs (Xu et al., 2024; Mitra et al., 2023). This dual utility highlights the importance of synthetic data generation as both a complementary and cost-efficient solution in the AD ecosystem.

C.2 Genarating Category Description Details

C.2.1 Prompt Details

As discussed in §4.3, we generate category descriptions to enhance LLM-based zero-shot AD. The prompt template used for generating category descriptions is shown in Table A13. It leverages the prompt techniques, including task information and CoT, as discussed in §B.1.

C.2.2 A Universal Component

LLM-generated category descriptions serve as a universal component that can be integrated into prompts to enhance any LLM-based task requiring category-specific information. In our study, we demonstrate its effectiveness in improving LLM-based zero-shot AD as shown in Table 2. Additionally, these descriptions can enhance LLM-based synthetic data generation similarly. This approach aligns with the Native Chain-of-Thought (NCoT) process (Wang et al., 2024a) used in OpenAI o1 (OpenAI, 2024c). Extending this idea, other datasets with distinct structures could inspire the development of task-specific universal components, enabling tailored augmentation strategies for diverse LLM-based applications.

D Additional Details for Case Study 3

D.1 Evaluation Protocol and Prompt Details

As discussed in §4.3, we utilize the information of both dataset and candidate models to achieve UMS. The prompt template used for generating category descriptions is shown in Table A13.

Importantly, we restrict our selection to two-step methods mentioned in §A.2, as the structural differences between end-to-end and two-step methods introduce additional complexities to an already challenging task.

Table A7: Performance comparison of AD baselines *with* and *without* LLM-generated synthetic data across five datasets. The better results for each detector are highlighted in bold. The performance may vary due to the embedding changes from the language model.

Training Set	**AG News**		**BBC News**		**IMDB Reviews**		**N24 News**		**SMS Spam**	
	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑
OpenAI + AE										
without $\mathcal{D}_{\text{synth}}$	0.5054	0.1189	0.6016	0.1309	0.5014	0.1665	0.7119	0.1681	0.5000	0.1020
with $\mathcal{D}_{\text{synth}}$	**0.7643**	**0.2612**	**0.8133**	**0.3422**	**0.5993**	**0.1997**	**0.7469**	**0.2040**	**0.5226**	**0.1064**
OpenAI + DeepSVDD										
without $\mathcal{D}_{\text{synth}}$	0.5171	0.1237	**0.6127**	**0.1415**	**0.5667**	**0.1969**	**0.6278**	**0.1511**	**0.6398**	**0.1479**
with $\mathcal{D}_{\text{synth}}$	**0.5480**	**0.1341**	0.5923	0.1291	0.5293	0.1802	0.6162	0.1364	0.3351	0.0707
OpenAI + ECOD										
without $\mathcal{D}_{\text{synth}}$	0.5014	0.1180	0.5623	0.1208	**0.5000**	**0.1661**	0.6202	**0.1311**	0.4078	0.0789
with $\mathcal{D}_{\text{synth}}$	**0.6673**	**0.1920**	**0.7490**	**0.2872**	0.4931	0.1545	**0.6213**	0.1310	**0.5000**	**0.1020**
OpenAI + IForest										
without $\mathcal{D}_{\text{synth}}$	0.6120	0.1620	**0.7102**	**0.1903**	**0.5788**	**0.1947**	0.5331	0.1010	**0.6386**	**0.1467**
with $\mathcal{D}_{\text{synth}}$	**0.6465**	**0.1905**	0.5684	0.1361	0.4591	0.1498	**0.6277**	**0.1381**	0.1996	0.0600
OpenAI + LOF										
without $\mathcal{D}_{\text{synth}}$	**0.6404**	**0.1661**	**0.7128**	**0.2565**	0.6759	**0.2485**	0.7179	0.2061	0.7582	0.2445
with $\mathcal{D}_{\text{synth}}$	0.4678	0.1174	0.6575	0.2073	**0.6785**	0.2401	**0.7369**	**0.2422**	**0.9329**	**0.4717**
OpenAI + SO_GAAL										
without $\mathcal{D}_{\text{synth}}$	0.5657	0.1324	0.3240	0.0770	**0.5388**	**0.1659**	**0.3351**	**0.0654**	**0.3953**	**0.0823**
with $\mathcal{D}_{\text{synth}}$	**0.5832**	**0.1556**	**0.5033**	**0.1031**	0.3311	0.1165	0.2344	0.0575	0.1032	0.0560
OpenAI + LUNAR										
without $\mathcal{D}_{\text{synth}}$	0.6527	0.2035	0.8554	0.4670	0.6546	0.2315	0.7879	0.2473	0.2305	0.0622
with $\mathcal{D}_{\text{synth}}$	**0.8590**	**0.4389**	**0.9093**	**0.6175**	**0.7011**	**0.2651**	**0.8025**	**0.2783**	**0.3394**	**0.0713**
OpenAI + VAE										
without $\mathcal{D}_{\text{synth}}$	0.6857	0.1842	0.7143	0.1816	0.5031	0.1670	0.6932	0.1698	**0.5000**	**0.1020**
with $\mathcal{D}_{\text{synth}}$	**0.7885**	**0.3124**	**0.7775**	**0.2973**	**0.6461**	**0.2240**	**0.7102**	**0.1705**	0.1110	0.0557

D.2 Failures on Popular LLMs

Despite the promising results achieved with GPT-o1-preview, widely used LLMs like GPT-4o and Llama 3.1 struggle with zero-shot UMS, frequently recommending the same model regardless of dataset context. This limitation highlights the need for enhanced reasoning abilities to better analyze dataset-specific requirements, model strengths and weaknesses, and their overall compatibility.

D.3 Complete Results

We present a radar chart in Figure 4 comparing the AD performance achieved by LLM-recommended models with the best performance and average performance. The detailed results with precise numerical values are provided in Table A6 for reference.

Table A8: Performance comparison of LLM-based detectors and baseline methods across five datasets. LLM-based detectors are evaluated under two settings as described in §3.2 with AUROC and AUPRC as the metrics (higher (↑), the better). The **best** results are highlighted in bold, the second-best results are double-underlined, and the third-best results are single-underlined.

Settings	**AG News**		**BBC News**		**IMDB Reviews**		**N24 News**		**SMS Spam**	
	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑	AUROC ↑	AUPRC ↑
					Llama 3.1 8B Instruct					
(1) with $\mathcal{C}_{\text{normal}}$	0.8226	0.4036	0.7910	0.3602	0.7373	0.3474	0.6267	0.1130	0.7558	0.2884
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.8754	0.3998	0.8612	0.3960	0.8625	0.4606	0.8784	0.3802	0.9487	0.6361
					GPT-4o					
(1) with $\mathcal{C}_{\text{normal}}$	**0.9332**	**0.7207**	0.9574	0.8432	0.9349	0.7823	0.7674	0.3252	0.7940	0.5568
(2) with $\mathcal{C}_{\text{normal}}, \mathcal{C}_{\text{anomaly}}$	0.9293	0.6310	**0.9919**	**0.9088**	**0.9668**	**0.8465**	**0.9902**	**0.9009**	**0.9862**	**0.8953**
Methods					**Baselines**					
CVDD	0.6046	0.1296	0.7221	0.2976	0.4895	0.1576	0.7507	0.2886	0.4782	0.0712
DATE	0.8120	0.3996	0.9030	0.5764	0.5185	0.1682	0.7493	0.2794	0.9398	0.6112
BERT + SO-GAAL	0.4489	0.1033	0.3099	0.0849	0.4663	0.1486	0.4135	0.0837	0.3328	0.0714
BERT + AE	0.7200	0.2232	0.8839	0.4274	0.4650	0.1479	0.5749	0.1255	0.6918	0.1914
BERT + DeepSVDD	0.6671	0.2160	0.5683	0.1328	0.4287	0.1387	0.4366	0.0798	0.5859	0.1178
BERT + ECOD	0.6318	0.1616	0.6912	0.2037	0.4282	0.1374	0.4969	0.0928	0.5606	0.1156
BERT + LOF	0.7432	0.2549	0.9320	0.6029	0.4959	0.1621	0.6703	0.1678	0.7190	0.1837
BERT + LUNAR	0.7694	0.2717	0.9260	0.5943	0.4687	0.1497	0.6284	0.1436	0.6953	0.1817
BERT + VAE	0.6773	0.1878	0.7409	0.2559	0.4398	0.1405	0.4949	0.0957	0.6082	0.1360
BERT + iForest	0.6124	0.1559	0.6847	0.2131	0.4420	0.1412	0.4724	0.0872	0.5053	0.0994
OpenAI + SO-GAAL	0.5945	0.1538	0.2359	0.0665	0.6201	0.3005	0.5043	0.0963	0.5671	0.1213
OpenAI + AE	0.8326	0.4022	0.9520	0.7485	0.6088	0.1969	0.7155	0.1984	0.5511	0.1030
OpenAI + DeepSVDD	0.4680	0.1062	0.5766	0.1288	0.6563	0.3278	0.6150	0.1297	0.3491	0.0721
OpenAI + ECOD	0.7638	0.3294	0.7224	0.2424	0.7366	0.5165	0.7342	0.2238	0.4317	0.0821
OpenAI + LOF	0.8905	0.5443	0.9558	0.7714	0.6156	0.2133	0.7806	0.2248	0.7862	0.2450
OpenAI + LUNAR	0.9226	0.6918	0.9732	0.8653	0.6474	0.2193	0.8320	0.4425	0.7189	0.1640
OpenAI + VAE	0.8144	0.3659	0.7250	0.2424	0.4515	0.1486	0.7418	0.2537	0.4259	0.0812
OpenAI + iForest	0.5213	0.1278	0.6064	0.1376	0.5064	0.1724	0.4944	0.0913	0.3751	0.0772

Table A9: LLM prompt template used for zero-shot AD in “Normal Only” setting discussed in §3.2. {normal_category_x} refers to the name of x_{th} normal category. {text} represents the test sample to be detected.

```
You are an intelligent and professional assistant that detects anomalies in text data.
## Task:
- Following the rules below, determine whether the given text sample is an anomaly. Provide a brief explanation
of your reasoning and assign an anomaly confidence score between 0 and 1.

## Categories:
- **{normal_category_1}**
- **{normal_category_2}**
- ...

## Rules:
1. **Anomaly Definition**:
   - A text sample is considered an **anomaly** if it does **not** belong to **any of the categories** listed
above.
2. **Scoring**:
   - Assign an anomaly confidence score between 0 and 1.
   - Use higher scores when you are highly confident in your decision.
   - Use lower scores when you are uncertain or think the text sample is **not** an anomaly.
3. **Step-by-step Reasoning** (Chain of Thought):
   - **Step 1**. Read the entire text sample carefully and understand it thoroughly.
   - **Step 2**. Analyze the text sample by comparing its content to each category listed in the "Categories"
section above, considering factors such as main topics, meanings, background, sentiments, etc.
   - **Step 3**. Determine which category the text sample **most closely aligns with**.
      - If it aligns with any category, it is **not** an anomaly.
      - If it does **not** align with any category, it is an anomaly.
   - **Step 4**. Assign an anomaly confidence score based on how confident you are that the text sample is
an anomaly.
4. **Additional Notes**:
   - A text sample may relate to multiple categories, but it should be classified into the **most relevant** one
in this task.
   - If you are uncertain whether the text sample **significantly aligns** with **any of the anomaly
category(ies)**, assume that it does **not**, which means it is **not** an anomaly.
5. **Response Format**:
   - Provide responses in a strict **JSON** format with the keys "reason" and "anomaly_score."
      - "reason": Your brief explanation of the reasoning in one to three sentences logically.
      - "anomaly_score": Your anomaly confidence score between 0 and 1.
   - Ensure the JSON output is correctly formatted, including correct placement of commas between key-value
pairs.
   - Add a backslash (\) before any double quotation marks (") within the values of JSON output for proper
parsing (i.e., from " to \"), and ensure that single quotation marks (’) are preserved without escaping.
Text sample:
"{text}"

Response in JSON format:
```

Table A10: LLM prompt template used for zero-shot AD in “Normal + Anomaly” setting discussed in §3.2. {normal_category_x} refers to the name of x_{th} normal category and {anomaly_category} refers to the name of anomaly category. {text} represents the test sample to be detected. The different part compared with the prompt in the “Normal Only” setting is marked in red.

```
You are an intelligent and professional assistant that detects anomalies in text data.
## Task:
- Following the rules below, determine whether the given text sample is an anomaly. Provide a brief explanation of your reasoning and assign an anomaly confidence score between 0 and 1.

## Categories:
### Normal Category(ies):
- **{normal_category_1}**
- **{normal_category_2}**
- ...
### Anomaly Category(ies):
- {anomaly_category}

## Rules:
1. **Anomaly Definition**:
  - A text sample is considered an **anomaly** if it belongs to the **anomaly category(ies)** rather than **any of the normal category(ies)** listed above.
2. **Scoring**:
  - Assign an anomaly confidence score between 0 and 1.
  - Use higher scores when you are highly confident in your decision.
  - Use lower scores when you are uncertain or think the text sample is **not** an anomaly.
3. **Step-by-step Reasoning** (Chain of Thought):
  - **Step 1**. Read the entire text sample carefully and understand it thoroughly.
  - **Step 2**. Analyze the text sample by comparing its content to each category listed in the "Categories" section above, considering factors such as main topics, meanings, background, sentiments, etc.
  - **Step 3**. Determine which category the text sample **most closely aligns with**.
    - If it **most closely aligns with** **any of the anomaly category(ies)**, it is an **anomaly**.
    - If it **most closely aligns with** **any of the normal category(ies)** instead, it is **not** an anomaly.
  - **Step 4**. Assign an anomaly confidence score based on how confident you are that the text sample is an anomaly.
4. **Additional Notes**:
  - A text sample may relate to multiple categories, but it should be classified into the **most relevant** one in this task.
  - If you are uncertain whether the text sample **significantly aligns** with **any of the anomaly category(ies)**, assume that it does **not**, which means it is **not** an anomaly.
5. **Response Format**:
  - Provide responses in a strict **JSON** format with the keys "reason" and "anomaly_score."
    - "reason": Your brief explanation of the reasoning in one to three sentences logically.
    - "anomaly_score": Your anomaly confidence score between 0 and 1.
  - Ensure the JSON output is correctly formatted, including correct placement of commas between key-value pairs.
  - Add a backslash (\) before any double quotation marks (") within the values of JSON output for proper parsing (i.e., from " to \"), and ensure that single quotation marks (') are preserved without escaping.
Text sample:
"{text}"

Response in JSON format:
```

Table A11: LLM prompt template used for keyword generation, which is the first step of generating synthetic samples as discussed in §4.2. {normal_category_x} refers to the name of x_{th} normal category. {name} and {original_task} can be found in Tab. A.1. {num_keyword_groups} set the number of keyword groups that LLM needs to generate for each category.

```
You are an intelligent and professional assistant that generates groups of keywords for given categories in a dataset.
## Task:
- Following the rules below, generate **exactly** {num_keyword_groups} unique keyword groups for **each given category** according to your understanding of the category (and its description).
- Each keyword group will be used to generate synthetic data for the corresponding category.

## Rules:
1. **Keyword Group Generation**:
   - For **each given category**, generate **exactly** {num_keyword_groups} keyword groups. Each group should contain exactly three keywords, with different levels of granularity: one broad/general, one intermediate, and one fine-grained.
   - Ensure that the three keywords in each group are thematically related to each other and align with the category's description.
   - Avoid redundancy or overly similar keywords across different groups.
   - Ensure that each group is unique and relevant to the key topics described in the category.
2. **Granularity**:
   - The first keyword should be broad/general, representing a high-level or overarching topic.
   - The second keyword should be intermediate, more specific than the first, but not overly narrow.
   - The third keyword should be fine-grained and specific, related to detailed subtopics or precise aspects of the category.
3. **Response Format**:
   - For each given category, provide the keyword groups as a list, where each entry is a group of three keywords (broad, intermediate, fine-grained).
   - Structure the response so that the key is the category name, and the value is a list of generated keyword groups.
   - Ensure the JSON output is properly formatted, including correct placement of commas between key-value pairs and no missing brackets.
   - Add a backslash (\) before any double quotation marks (") within the values of JSON output for proper parsing (i.e., from " to \"), and ensure that single quotation marks (') are preserved without escaping.

The "{name}" dataset's original task is {original_task}. It contains the following category(ies):
{normal_category_1}
{normal_category_2}
...

Response in JSON format:
```

Table A12: LLM prompt template used for sample generation, which is the second step of generating synthetic samples as discussed in §4.2. We generate a single synthetic sample per keyword group. {keyword_group[i]} refers to $(i+1)_{th}$ granularity level's keyword in this keyword group. {category} represents the name of the corresponding category for this keyword group.

```
You are an intelligent and professional assistant that generates a synthetic text sample based on a group of 3 keywords with different levels of granularity.
## Task:
- Generate a synthetic text sample that incorporates the provided group of 3 keywords (broad, intermediate, and fine-grained) listed below.
- The generated sample should align with the meanings and themes suggested by the keywords provided.

## Rules:
1. **Sample Characteristics**:
   - Generate a synthetic text sample that naturally incorporates the three provided keywords (broad, intermediate, and fine-grained).
   - Ensure that the text sample is coherent and contextually relevant to the themes suggested by the keywords.
2. **Keyword Usage**:
   - The three keywords must appear naturally within the content.
   - Ensure that the broad keyword sets the overall context, the intermediate keyword refines the discussion, and the fine-grained keyword offers more detailed insight into a specific subtopic.
3. **Response Format**:
   - Provide the generated sample as a single string response representing the text sample.
   - Ensure the output is in a readable format.
   - Do not include any additional messages or commentary.
   - Add a backslash (\) before any double quotation marks (") within the values of JSON output for proper parsing (i.e., from " to \"), and ensure that single quotation marks (') are preserved without escaping.

The "{name}" dataset's original task is {original_task}. The category is "{category}", and the group of keywords to use is:
- Broad: {keyword_group[0]}
- Intermediate: {keyword_group[1]}
- Fine-grained: {keyword_group[2]}

Response in JSON format:
```

Table A13: LLM prompt template for generating category descriptions discussed in §4.3. {normal_category_x} refers to the name of x_{th} normal category and {anomaly_category} refers to the name of anomaly category. {name} and {original_task} can be found in Tab. A.1.

You are an intelligent and professional assistant that generates descriptions for given categories in a text dataset.
Task:
- Following the rules below, generate detailed textual descriptions that explain the main characteristics, typical topics, and common examples for each given category.

Rules:
1. For each category, provide a continuous, coherent description in a single paragraph that includes:
   - **Definition or overview**: Start by briefly defining or describing the category in one to two sentences. If you list multiple aspects or features in the definition (such as related fields or industries), ensure you append expressions like "etc." or "and so on" to indicate that the list is not exhaustive.
   - **Main topics or subjects**: Highlight the typical topics or subjects covered by this category. Ensure that you use phrases like "etc." or "and so on" at the end of each list to indicate that the list is not exhaustive.
- **Relevant examples**: Mention examples of content that belong to this category. Also, use expressions like "etc." or "and so on" at the end of the list to show that these are illustrative, not exhaustive.
2. Use **step-by-step reasoning** to ensure the descriptions are logical and clear.
3. Each description should be clear, coherent, and helpful for someone unfamiliar with the dataset and the task.
4. Always append phrases like "etc." or "and so on" to lists or enumerations of examples, topics, or aspects, **including the definition part**.
5. Response Format:
   - Provide a response where each key is the category name, and the value is the corresponding description as a continuous paragraph.
   - Ensure the JSON output is correctly formatted, including correct placement of commas between key-value pairs.
   - Add a backslash (\) before any double quotation marks (") within the values of JSON output for proper parsing (i.e., from " to \"), and ensure that single quotation marks (') are preserved without escaping.

The "{name}" dataset's original task is {origianl_task}. It contains the following categories:
{normal_category_1}
{normal_category_2}
...
{anomaly_category}

Response in JSON format:

Table A14: LLM prompt template used for UMS discussed in §5. {normal_category_x} refers to the name of x_{th} normal category and {anomaly_category} refers to anomaly one. We randomly select examples from the training set for both normal and anomaly data, denoted as {normal_text} and {anomaly_text}. {name}, {size} (i.e., # of test set), and {original_task} can be found in Tab. A.1. {avg_len}, {max_len}, {min_len}, and {std_len} are statistics of datasets as shown in Tab. A1. {abstract} is the abstract in the published paper of each model.

```
You are an expert in model selection for anomaly detection on text datasets.

## Task:
- Given the information of a dataset and a set of models, select the model you believe will achieve the best performance for detecting anomalies in this dataset. Provide a brief explanation of your choice.

## Dataset Information:
- Dataset Name: {name}
- Dataset Size: {size}
- Background: This dataset is originally for {original_task}.
- Data Structure: Textual data with multiple categories. One category is considered anomalous, while the others are normal.
  - Normal Category(ies): {normal_category_1}, {normal_category_2}
    - An Example: {normal_text}
  - Anomaly Category: {anomaly_category}
    - An Example: {anomaly_text}
- Text Length Statistics:
  - Average Length: {avg_len}
  - Maximum Length: {max_len}
  - Minimum Length: {min_len}
  - Standard Deviation: {std_len}

## Model Information:
- Models utilize language models to generate embeddings and feed the embeddings into the models.
- We provide the abstracts of the papers that introduce the models for your reference.
### Model Options:
- AutoEncoder (AE): {abstract} (Aggarwal, 2015)
- Deep Support Vector Data Description (DeepSVDD): {abstract} (Ruff et al., 2018)
- Empirical-Cumulative-Distribution-Based Outlier Detection (ECOD):{abstract} (Li et al., 2022)
- Isolation Forest (IForest): {abstract} (Liu et al., 2008)
- Local Outlier Factor (LOF): {abstract} (Breunig et al., 2000)
- Unifying Local Outlier Detection Methods via Graph Neural Networks (LUNAR): {abstract} (Goodge et al., 2022)
- Single-Objective Generative Adversarial Active Learning (SO-GAAL): {abstract} (Liu et al., 2019)
- Variational AutoEncoder (VAE): {abstract} (Kingma and Welling, 2014)
### Embedding Options:
- Bidirectional Encoder Representations from Transformers (BERT): {abstract} (Kenton and Toutanova, 2019)
- "text-embedding-3-large" from OpenAI (referred to as OpenAI): {abstract} (OpenAI, 2024b)

## Rules:
1. Availabel options include "BERT+AE", "BERT+DeepSVDD", "BERT+ECOD", "BERT+iForest", "BERT+LOF", "BERT+LUNAR", "BERT+SO-GAAL", "BERT+VAE", "OpenAI+AE", "OpenAI+DeepSVDD", "OpenAI+ECOD", "OpenAI+iForest", "OpenAI+LOF", "OpenAI+LUNAR", "OpenAI+SO-GAAL", "OpenAI+VAE."
2. Treat all models equally and evaluate them based on their compatibility with the dataset characteristics and the anomaly detection task.
3. Response Format:
  - Provide responses in a strict **JSON** format with the keys "reason" and "choice."
    - "reason": Your explanation of the reasoning.
    - "choice": The model you have selected for anomaly detection in this dataset.

Response in JSON format:
```