Targeting the Needle, Ignoring the Haystack: Anchoring Crucial Cues for Evolving Scam Call Detection via an LLM-Assisted Classifier

Tong Wu[1], Qinliang Su[1,2]*, Jianxing Yu[3], Bo Liang[4], Minhua Huang[4]
[1]School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China
[2]Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
[3]School of Artificial Intelligence, Sun Yat-sen University, Guangdong, China
[4]China Mobile Internet Company Ltd.
{wutong85@mail2, suqliang@mail, yujx26@mail}.sysu.edu.cn
{huangminhua, liangbo}@cmic.chinamobile.com

Abstract

Automatic detection of fraudulent voice calls is essential for online service platforms but faces significant challenges due to the scarcity of labeled data and the continuous evolution of conversational contexts. Standard supervised methods often fail to generalize, as they tend to overfit to variable background narratives rather than capturing the core deceptive intent. In this paper, we propose a lightweight framework that anchors detection on Semantic Primitives, a set of stable, interpretable evidentiary cues derived from expert knowledge. Our approach decomposes the fraud detection task into two distinct stages: identifying the presence of these predefined semantic signals within the transcript, and deriving a final verdict through a logical combination of the detected cues. By explicitly prioritizing stable evidence over diverse conversational noise, this framework ensures that decisions are based on verifiable fraud tactics rather than spurious correlations. Experimental results demonstrate that our method achieves superior robustness and efficiency compared to traditional baselines, particularly in scenarios with shifting service contexts.

1 Introduction

With the rapid advancement of communication technologies, voice calls have become a common channel for remote interaction. However, this convenience has also created new avenues for telecom fraud. Beyond traditional personal communication, voice calls are now widely used on online platforms for daily services (e.g., recruitment and package delivery). Unfortunately, the leakage of phone numbers has led to a growing number of scam calls on these platforms. For example, fraudsters may impersonate platform staff and contact users under various pretexts like job screening or service confirmation to defraud them. By exploiting users' trust in the platform, such schemes often achieve a high success rate. Therefore, deploying automated fraud-detection algorithms on online service platforms has become essential.

Some recent work has been devoted to developing low-complexity methods for scam call detection. For instance, Li et al. (2024) trained a RoBERTa-based classifier using a dual-loss framework to efficiently detect scam calls, while Oyeyemi and Ojo (2024) combined BERT (Devlin et al., 2019) with Naive Bayes for SMS spam detection. Despite their reported success, the robustness of these methods typically hinges on sufficient annotations for both scam and normal calls. But due to the high cost of expert annotation and the low occurrence of scam calls, available training data are often limited in size. More seriously, fraudsters often conceal their malicious intents within highly diverse and evolving contexts. Fig. 1 illustrates two examples of scam calls that try to deceive job seekers into accepting fake paid training on an online recruitment platform. In the first example, fraudsters conceal their intent by initiating a conversation that pretends to provide job opportunities in the software industry, while in the second one, as live streaming becomes popular, fraudsters swiftly change their focus to live streaming. Clearly, even on a single recruitment platform, the contents of calls could be extremely diverse, not to mention that they still constantly evolve over time. If the classifier is directly trained, especially when the annotation is not sufficient, it may easily discover some spurious correlation, *e.g.*, simply attributing scams to "Software Company" only because the training data contains some scam calls mentioning "software company".

Given the difficulty of collecting sufficient scam calls that cover all possible scenarios, recent approaches(Jiang, 2024; Shen et al., 2025a,b; Singh et al., 2025) have focused on leveraging the powerful reasoning capabilities of Large Language Models (LLMs) for zero-shot analysis. While these

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 20497–20513

Case 1: We are from xxx Software Technology Company, specializing in software testing. Are you currently looking for new career opportunities?……(Introduction to Software Testing)…… **Since you don't have experience, we will have a senior instructor mentor you one-on-one during the initial period**… For four consecutive years, the internet industry has ranked as the highest-paying sector in the country……(More Background)……

Case 2: We are from xxx Company and we focus on **livestream sales**. Would you consider this?……(Introduction to the livestreaming industry)……**It's okay if you haven't done it before, we provide training**……Are you interested? Our salary ranges from ￥4,000 to ￥6,000……(Details on benefits)……

Figure 1: Two examples of scam calls from recruitment platforms.

methods achieve promising performance in scenarios with only a few annotations, the high inference cost of LLMs limits their practicality for large-scale or real-time deployment. In real-world platforms that handle millions of calls every day, leveraging LLMs to analyze the content of every single call would result in prohibitive computational overhead. Therefore, lightweight fraud detection methods based on traditional classifiers (e.g., BERT-based models) remain important, as they can efficiently process the vast majority of routine service traffic, reserving expensive LLM inference solely for high-risk cases, or can even be deployed to detect scam calls in real time.

To develop a lightweight classifier under insufficient annotation, we observe that while calls' contexts are highly diverse and dynamic, the core tactics employed by fraudsters remain largely stable. As shown in the examples of Fig. 1, although conversation contexts could vary a lot, the scam tactic of "*inducing job seekers to participate in job-related training programs*" remains unchanged. Therefore, we can detect this type of scams by seeking to identify this specific semantic evidence, without being distracted by irrelevant contexts. This idea also aligns with the way of how human experts screening scam calls: instead of carefully examining every word in a lengthy transcript, experts simply scan the transcripts and identify specific signals to make a judgment.

Mimicking this "search-and-verify" process, we design a framework to explicitly anchor detection on crucial evidence. To this end, we first introduce Semantic Primitives as semantic prototypes for different categories of evidence signals. Building on these pre-defined primitives, we decompose the fraud detection task into two easier sub-tasks: a primitive recognition task to identify the presence of predefined semantic cues within the transcript, and a decision task that derives the final result based on the status of these primitives. To realize the identification task, we train a lightweight detector using a dataset that is curated with the help of LLM to identify the presence of primitives. For the decision task, we derive the final verdict by performing logical operations on these detected signals according to prior knowledge. We evaluated our framework on a real-world scam call dataset collected from an online recruitment platform and a publicly available synthetic dataset (Ma et al., 2025). Experimental results demonstrate that while traditional baselines suffer catastrophic performance degradation in evolving and diverse conversational contexts, our framework maintains remarkable robustness.

2 Related Work

The increasing complexity of telecom fraud (Al Saidat et al., 2024; Kou et al., 2004) has driven the development of diverse detection paradigms. In behavioral analysis, Graph Neural Networks (GNNs) have gained attention as a powerful modeling approach (Hu et al., 2023; Liu et al., 2021; Ying et al., 2018; Tseng et al., 2015). However, such methods suffer from inherent time-lag issues (Ying et al., 2018), as they depend on sufficient historical interactions to achieve reliable performance. For textual scam detection, early deep learning methods adopted BiLSTM with attention mechanisms (Xu et al., 2022) or hybrid CNN–LSTM architectures (Ghourabi et al., 2020) to capture keyword importance and sequential dependencies. To better capture semantic nuances, subsequent research has shifted towards Pre-trained Language Models (PLMs). Oyeyemi and Ojo (2024) and Jain et al. (2025) employed BERT embeddings (Devlin et al., 2019) to achieve high classification accuracy, while Chen and Chen (2024) introduced Efficient Additive Attention to reduce computational overhead. To further improve robustness in complex scenarios, dual-loss frameworks (Li et al., 2024) with RoBERTa (Liu et al., 2019) have also been explored. Nevertheless, these approaches primarily depend on the model to autonomously discover discriminative patterns from training data. More recently, Large Language Models (LLMs) have demonstrated strong potential in identifying sophisticated scams (Jiang, 2024; Shen et al., 2025a,b; Singh et al., 2025; Ma et al., 2025) due to their su-

perior reasoning capabilities (Brown et al., 2020). However, their high inference cost makes them impractical for processing massive volumes of daily calls. As a result, robust and efficient lightweight models remain crucial for large-scale real-world deployment.

3 Problem Description

In this study, we consider a fraud detection setting with the availability of a small training dataset $\mathcal{D}_{\text{train}} = \{(d_i, y_i)\}_{i=1}^{N}$, where d_i represents a call transcript and y_i indicates its corresponding label (i.e., scam or normal). Unlike general text classification tasks, in which discriminative semantics are densely distributed throughout the input d_i, we focus on telecom fraud scenarios with key signals sparsely distributed within the transcripts, such as recruitment scenarios. Under these scenarios, a call transcript d can be formulated as a collection of text spans, comprising both contexts (c) and critical evidence (e):

$$d = [c_1, e_1, c_2, \ldots, c_n, e_2, c_{n+1}, \ldots]. \quad (1)$$

Here, $\{c_k\}$ denotes context spans consisting of routine dialogue, while $\{e_k\}$ represents specific evidence spans that are decisive for determining the nature of the call. In practice, as fraudsters frequently conceal their malicious intent within normal business interactions, the context spans $\{c_k\}$ typically dominate the length of the transcript d and exhibit extreme diversity across different scenarios (e.g., technology company recruitment or e-commerce platform recruitment). In contrast, the evidence $\{e_k\}$ may consist of only a few brief segments. For instance, considering a typical scam pattern characterized by "*claiming abundant job openings without mentioning specific job titles*", if a call includes a short segment mentioning "*abundant job openings*", this would raise suspicion. If this call further contains a span describing "*specific job titles*," the suspicion, however, could be dimmed. Therefore, the nature of calls is largely decided by some very short spans $\{e_k\}$, while being irrelevant to the lengthy contexts.

Classifiers that are directly trained to fit the labels often fail to yield such an ideal decision function. Due to the diversity of conversation content, the training dataset is almost impossible to cover all scenarios, especially considering the continuously evolving characteristic of calls. Given the dominant length of contexts in calls, classifiers inevitably get stuck at finding some spurious correlation within the irrelevant contexts, while failing to discover the true causes (i.e., the decisive short spans), rendering it poor at generalizing to detect unseen call transcripts.

4 Methodology

4.1 Overall Framework

To design a framework capable of efficiently identifying scam calls, we draw inspiration from the key observation that while the context $\{c_k\}$ exhibits high variability, the core fraud tactics embedded in $\{e_k\}$ remain relatively stable over time. Motivated by this insight, we propose to explicitly summarize these evidence spans $\{e_k\}$ into a set of **Semantic Primitives**, with each primitive representing the semantic prototype of evidence spans that share similar semantics. For instance, for the recruitment scams featuring the claim of "*abundant job openings without mentioning specific job titles*", although the same meaning could be expressed in numerous ways in a call, we only need to define two primitives to capture their core semantics: p_1 = "*mention of specific job titles*" and p_2 = "*claim of abundant job openings*". Based on these primitives, the type of scam mentioned above can be explicitly captured by a Boolean logical operation on the primitives p_1 and p_2 as $\neg p_1 \wedge p_2$, which means the call did not mention specific job titles but claimed there were abundant job openings.

Building on this idea, we propose a two-stage framework. Formally, letting $\mathcal{P} = \{p_1, p_2, \ldots, p_K\}$ denote the set of K predefined semantic primitives, we first propose to train a lightweight detection model $\mathcal{F}(\cdot)$ to scan the transcript d and identify the existence status of the primitives in $\mathcal{P}$:

$$\mathbf{z} = \mathcal{F}(d), \quad (2)$$

where $\mathbf{z} \in \{0, 1\}^K$ is the *Primitive Status Vector*, and the k-th entry z_k indicates the presence (1) or absence (0) of primitive p_k.

Then, we derive the final verdict on whether the call is a scam by applying a logic decision function $Logic(\cdot)$ to the primitive-status vector $\mathbf{z}$:

$$y = Logic(\mathbf{z}). \quad (3)$$

Unlike the learnable detection model $\mathcal{F}(\cdot)$, $Logic(\cdot)$ is a fixed mapping (e.g., Boolean logic clause) that yields the final decision by performing

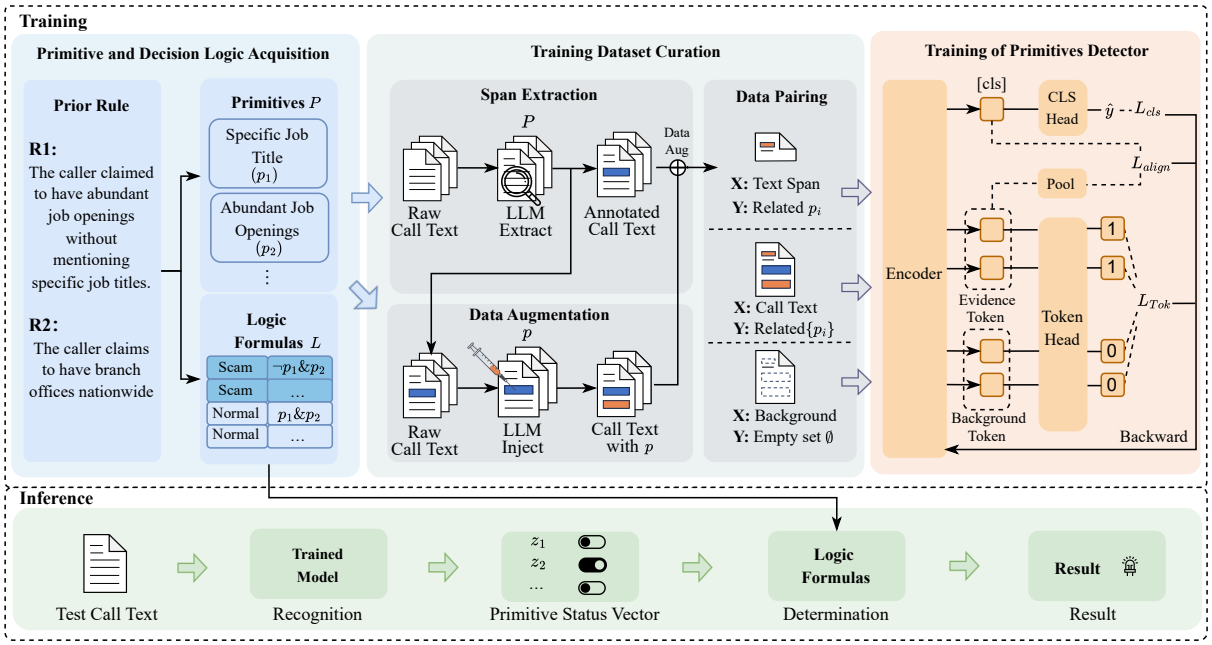

Figure 2: Architecture of the proposed framework.

logical operation on the status of primitives in $\mathcal{P}$. For instance, in the example mentioned above, the logic mapping $Logic(\mathbf{z}) = \neg z_1 \wedge z_2$. By separating the decision process into explicit semantic recognition and logic reasoning, this approach effectively relieves the model from the burden of implicitly deriving complex decision logic.

4.2 Primitive and Decision Logic Acquisition

For the domain-specific set of semantic primitives $\mathcal{P}$ and the corresponding decision logic $Logic(\cdot)$ stated above, there are generally two ways to obtain them. The first approach relies on manual definition, in which domain experts analyze common fraud patterns to identify a set of semantic primitives and formalize them into rigorous logical expressions. Second, in scenarios where guidance rules are available for judging whether a call is a scam, we can also prompt an LLM to automatically parse these rules into the required primitives and the corresponding logical decision function. Further details on the prompting strategy and implementation are provided in Appendix C.1. Through this acquisition phase, abstract domain knowledge is transformed into a structured representation that can be directly used by the proposed framework, with specific semantic primitives and decision logic detailed in Appendix B.

4.3 Training Data Curation

To develop a lightweight detection model $\mathcal{F}(\cdot)$ capable of discerning the presence of semantic primitives within call transcripts, we construct a dataset to provide fine-grained supervision for its training. This curation process comprises three stages: (1) Primitive-Relevant Span Extraction, (2) Data Augmentation via Primitive Injection, and (3) Transcript-Label Pairs Construction.

4.3.1 Primitive-Relevant Span Extraction

To curate the training dataset, we first propose a method to pinpoint the text spans that are semantically similar to the primitives in $\mathcal{P}$. Specifically, for each transcript $d \in \mathcal{D}_{\text{train}}$ and each candidate primitive $p \in \mathcal{P}$, we design a retrieval prompt T_{ext} to guide the LLM in identifying minimal relevant text spans in d that provide evidence for p. The extraction process can be formulated as:

$$\mathcal{S}_{d,p} = LLM(d, p; T_{\text{ext}}), \tag{4}$$

where $\mathcal{S}_{d,p} = \{s_1, s_2, \ldots, s_m\}$ is the set of extracted evidence spans supporting p in transcript d. Based on the extraction results, for a given transcript d, we collect the set of primitives supported by any extracted evidence, denoted as $\mathcal{P}_d = \{p \in \mathcal{P} \mid \mathcal{S}_{d,p} \neq \emptyset\}$, and gather all associated evidence spans into $\mathcal{S}_d = \bigcup_{p \in \mathcal{P}_d} \mathcal{S}_{d,p}$.

4.3.2 Data Augmentation via Primitive Injection

To obtain a diverse set of call transcripts containing fraud-related primitives, we employ an LLM-driven Data Augmentation strategy to construct an augmented dataset $\mathcal{D}_{\text{aug}}$. Specifically, this process involves two sequential steps. First, we perform primitive paraphrasing to enrich linguistic diversity. We employ a dedicated prompt T_{para} to guide the LLM in expanding each target primitive p into K variations. These variations are semantically equivalent but stylistically distinct, denoted as $\mathcal{V}_p = \{v_1, v_2, \ldots, v_K\}$:

$$\mathcal{V}_p = LLM(p; T_{\text{para}}). \tag{5}$$

Subsequently, we perform semantic injection to embed the synthesized variations within the background d. To achieve this, we sample a background transcript d from the training set $\mathcal{D}_{\text{train}}$ and a paraphrase $v \in \mathcal{V}_p$, then prompt the LLM with an injection prompt T_{inject} which directs the model to integrate v into an appropriate position while strictly preserving the original structure of d. The generation of the augmented transcript d_{aug} can be expressed as:

$$d_{\text{aug}} = LLM(d, v; T_{\text{inject}}). \tag{6}$$

Notably, as d_{aug} is synthesized through this controlled editing process, the corresponding primitive set $\mathcal{P}_{d_{\text{aug}}}$ and evidence spans $\mathcal{S}_{d_{\text{aug}}}$ are known by construction. This eliminates the need for an additional evidence extraction pass. Finally, we collect all generated instances d_{aug} and construct an augmented dataset $\mathcal{D}_{\text{aug}}$.

4.3.3 Transcript-Label Pairs Construction

To train the lightweight detection model $\mathcal{F}(\cdot)$, we construct a unified training set $\mathcal{D}_{\text{final}} = \{(x_i, \mathbf{z}_i)\}_{i=1}^{N}$ containing pairs of input text x and a primitive status vector $\mathbf{z} \in \{0, 1\}^K$. For each transcript $d \in \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{aug}}$, since its associated evidence spans $\mathcal{S}_d$ and primitives $\mathcal{P}_d$ are known, we now use them to construct three distinct training pairs:

(1) *Global Context Instances*: To enable primitive detection within the full call transcript, we directly use every document $d \in \mathcal{D}_{\text{aug}} \cup \mathcal{D}_{\text{train}}$ and its associated primitive set $\mathcal{P}_d$ to construct a pair $(d, \mathbf{z}_d)$, where $\mathbf{z}_d$ is a multi-hot vector that indicates the presence of the primitive from the set $\mathcal{P}_d$.

(2) *Background Negative Instances*: To explicitly prevent the model from overfitting to frequent but irrelevant context, we construct the pair $(d_{\text{bg}}, \mathbf{0})$, where $d_{\text{bg}} = d \setminus \mathcal{S}_d$ denotes the text that has the evidence spans removed, and $\mathbf{0}$ is a K-dimensional all-zero vector indicating the absence of any primitives from $\mathcal{P}$.

(3) *Local Evidence Instances*: To ensure the model recognizes core semantic cues alone, for each document d and each evidence span $s \in \mathcal{S}_d$ that supports a primitive p_k, we formulate the pair $(s, \mathbf{1}_k)$, where $\mathbf{1}_k$ is a one-hot vector with the k-th entry set to 1.

4.4 Training of Primitives Detector

To efficiently judge the existence of any semantic primitives from $\mathcal{P}$ within lengthy call transcripts, we propose training a classifier by optimizing three mutually cooperative objectives.

We first use the curated dataset $\mathcal{D}_{\text{final}}$ to train a multi-label binary classifier. Specifically, for each primitive p_k, we build a binary classifier by adding a linear sigmoid layer on top of the [CLS] embedding $\mathbf{h}_{\text{CLS}}$ from BERT; that is, $\hat{z}_k = \sigma(\mathbf{w}_k^T \mathbf{h}_{\text{CLS}})$. Then, for every $(d, \mathbf{z}) \in \mathcal{D}_{\text{final}}$, we minimize the following loss to recognize the existence of each primitive:

$$L_{\text{cls}} = -\sum_{k=1}^{K} \left[z_k \log(\hat{z}_k) + (1 - z_k) \log(1 - \hat{z}_k) \right], \tag{7}$$

where z_k denotes the k-th element of $\mathbf{z}$.

In addition, to explicitly guide the model to pinpoint the primitive-relevant spans, we incorporate a token-level evidence identification task sharing the same encoder as the classifier above. Specifically, we build a token-wise classifier by adding a linear sigmoid layer on top of the final hidden representation $\mathbf{h}_i$ of each token, formulated as $\hat{t}_i = \sigma(\mathbf{w}_{\text{token}}^T \mathbf{h}_i)$. Then, for any input x associated with evidence set $\mathcal{S}_x$, we define a target binary sequence $\mathbf{t}$, where $t_i = 1$ if the i-th token belongs to any span in $\mathcal{S}_x$, and 0 otherwise. Then, we minimize the token-wise classification loss:

$$L_{\text{token}} = -\sum_{i \in x} \left[t_i \log(\hat{t}_i) + (1 - t_i) \log(1 - \hat{t}_i) \right]. \tag{8}$$

Finally, specifically for Global Context Instances, we encourage consistency between the global verdict and local evidence by aligning the global vector $\mathbf{h}_{\text{cls}}$ with an evidence representation $\mathbf{h}_{\text{local}}$ derived by pooling the representations of

identified evidence tokens. This alignment is optimized via a cosine-based loss:

$$L_{\text{align}} = 1 - \cos\left(\mathbf{h}_{\text{cls}}, \mathbf{h}_{\text{local}}\right). \quad (9)$$

The overall objective aggregates these terms to jointly optimize for detection accuracy:

$$L_{\text{total}} = L_{\text{cls}} + L_{\text{token}} + \lambda L_{\text{align}}, \quad (10)$$

where λ is a hyperparameter balancing the impact of the alignment constraint.

5 Experiment

5.1 Experimental Setups

Datasets and Metrics To evaluate the performance of our method, we conduct experiments on two Chinese call transcript datasets: (1) **SynthScamCall** (Ma et al., 2025), a publicly available synthetic dataset constructed based on real call patterns; and (2) **RealCall**, a collection of real-world call transcripts curated and provided by an online recruitment platform. Table 1 summarizes the statistics of these datasets. More details regarding the datasets are provided in Appendix D.1.

Unlike conventional benchmarks, these datasets are characterized by a strict temporal split between the training and testing sets, thereby resulting in **highly diverse** and **markedly different** surface-topic distributions across the two sets. This experimental setup allows for rigorous testing of whether a model is merely memorizing specific topics and background patterns (e.g., specific company names) or is relying on truly discriminative evidence. Overall, these datasets reflect the inherent challenges of real-world service scenarios: (1) normal calls are much more frequent than fraudulent ones; (2) critical evidence is often subtle and sparsely distributed across noisy conversations; and (3) background contexts are highly dynamic and constantly evolving over time.

Baselines To provide a comprehensive evaluation, we compare our approach against a diverse set of baseline methods:

(1)*Weakly Supervised Learning:* Weakly supervised learning (Maron and Lozano-Pérez, 1997) addresses scenarios where supervision is limited to sample-level labels, and the model must both pinpoint task-relevant evidence during training and perform evidence-driven inference without access to explicit evidence annotations. Given that only sample-level labels of call transcripts are available while the precise locations of evidence remain unknown, we employ this category of methods for comparison. Among these methods, Multiple Instance Learning **(MIL)** is the most representative paradigm. In the MIL implementation (Ilse et al., 2018), we segment each call transcript into a series of text spans and optimize the model to discover suspicious fragments within them. Additionally, we include **RNP** (Lei et al., 2016) for comparison, which uses the REINFORCE algorithm to explore evidence during training.

Dataset	$\mathcal{D}_{\text{train}}$		$\mathcal{D}_{\text{test}}$	
	#Scam	#Call	#Scam	#Call
SynthScamCall	592	6,962	223	2,293
RealCall	1,141	5,236	845	18,845

Table 1: Statistics of the **RealCall** and **SynthScamCall** datasets. #Scam denotes the number of scam call transcripts and #Call denotes the number of call transcripts.

(2) *Adversarial Training*: Since adversarial training is effective in suppressing background variation and learning domain-invariant representations, we introduce this category of methods for comparison. Specifically, we follow the method described in **ELS** (Zhang et al., 2023) as the representative for this class of methods.

(3) *Generative Data Augmentation*: We further explore sample generation to augment the training of other baseline models. In our experiment, we include **ZeroGen** (Ye et al., 2022a), which converts task descriptions into prompts for LLMs to generate labeled samples, and **ProGen** (Ye et al., 2022b), which refines generation via feedback derived specifically from a standard classifier.

Implementation Details For our main method, we use the following configuration. We perform data augmentation with 6 target primitives, each expanded into $K = 50$ semantic variations. Every variation is injected into two normal transcripts, resulting in 600 augmented instances. During training, we use a batch size of 64 and a learning rate of 2×10^{-5}, with λ set to 0.3. For MIL, we treat the entire call transcript as a bag and segment the text into fixed-length windows of 50 tokens, which serve as the instances for training. All methods utilize Bert-base-chinese as the text encoder. Further details regarding the experimental setup are provided in Appendix D.

Setting	LLM	Method	SynthScamCall			RealCall		
			Precision	Recall	F1-Score	Precision	Recall	F1-Score
Baselines (Only Ori-Data)	∼	Finetuned-BERT	0.128	0.860	0.223	0.041	0.637	0.077
		MIL	0.136	0.639	0.224	0.049	0.013	0.021
		RNP	0.110	0.675	0.189	0.028	0.211	0.049
Baselines (With Gen-Data)	Qwen3-max	Zerogen	0.155	0.855	0.262	0.043	0.515	0.079
		ProGen	0.123	**0.927**	0.217	0.040	0.278	0.070
		Zerogen+ELS	0.108	0.720	0.188	0.056	0.640	0.103
		Zerogen+MIL	0.157	0.779	0.261	0.084	0.434	0.141
		Zerogen+RNP	0.189	0.810	0.306	0.056	0.573	0.102
	Deepseek-v3	Zerogen	0.141	0.873	0.243	0.053	0.652	0.098
		ProGen	0.157	0.716	0.258	0.044	0.142	0.067
		Zerogen+ELS	0.101	0.734	0.178	0.054	0.453	0.096
		Zerogen+MIL	0.147	0.756	0.246	0.095	0.330	0.148
		Zerogen+RNP	0.164	0.864	0.276	0.044	0.443	0.080
Primitive-Guided Framework	Qwen3-max	Ours	0.624	0.846	**0.718**	**0.511**	0.664	**0.578**
	Deepseek-v3	Ours	**0.638**	0.810	0.714	0.485	**0.679**	0.566

Table 2: Performance comparison of our method against baselines on SynthScamCall and RealCall datasets. The best results are highlighted in bold.

5.2 Experimental Results

The overall performance of various baseline combinations and our proposed method is presented in Table 2. From the table, we can see that standard BERT-based classifiers suffer from severe performance degradation on both datasets, even when equipped with ELS for adversarial training. Despite the high recall rate observed, they show extremely low precision, indicating that the classifiers fail to capture sparse key evidence from long and diverse contexts.

Although Weakly Supervised Learning methods attempt to locate the key evidence, they still fail to yield significant improvements. For the MIL method, the marginal improvement can be attributed to two factors. The first factor is that MIL implicitly assumes the label is solely determined by the presence of suspicious segments. However, this is not sufficient because the final verdict depends on their logic combination. The other factor is that the diverse and long contexts are still easy to make the MIL classifier discover some spurious correlations that result in the scams. As for RNP, it fails primarily because the vast search space induced by the lengthy contexts makes the REINFORCE algorithm highly unstable and difficult to converge. As for the Generative Data Augmentation strategies, although they provide a slight performance improvement by increasing the sample size, since they do not resolve the fundamental interference of the dominant background context, their overall gains remain limited.

Method	SynthScamCall	RealCall
Rule-Guided CoT	0.84	0.60
Ours	0.71	0.57

Table 3: Performance comparison between our lightweight framework and the LLM-based Rule-Guided CoT.

In contrast to these baselines, our proposed framework shows significant performance improvement across both datasets. By explicitly anchoring detection on pre-defined Semantic Primitives, our method enables the framework to prioritize the screening of critical evidence and thus reduce the reliance on spurious shortcuts, leading to decisions that are more consistently aligned with the stable fraud tactics. In addition to comparing with these classifier-based methods, we also compare our lightweight method with the state-of-the-art LLM-based scam detector (Ma et al., 2025), with the experimental results shown in Table 3. From the table, we can see that although our method is much cheaper than the LLM-based method at the inference stage, our method still achieves a similar performance as it. This again demonstrates the effectiveness of our proposed approach by detecting the existence of critical primitives and then using logical reasoning to yield the final decision.

5.3 Ablation Study

Ablation of Framework Components We conduct an ablation study to assess the contribution

Model Setting	SynthScamCall	RealCall
Basic Two-Stage	0.57	0.48
+ Data Augmentation ($\mathcal{D}_{\text{aug}}$)	0.63	0.54
+ Token-Level Task (L_{token})	0.66	0.54
+ Alignment Task (L_{align})	**0.71**	**0.57**

Table 4: Ablation study on the contribution of each component in our framework.

Supervision Setting	SynthScamCall	RealCall
w/o Extraction	0.45	0.49
w/ Extraction (Ours)	**0.63**	**0.54**

Table 5: Ablation study on the impact of Primitive-Relevant Span Extraction.

of each component in our framework, with the results summarized in Table 4. To begin with, we evaluate a Basic Two-Stage baseline to validate the structural advantage of our framework. In this baseline, a standard classifier is trained on the curated original dataset to identify primitives, and the final decision is derived using the predefined logic. Despite its simplicity, this baseline achieves competitive performance, demonstrating the effectiveness of the basic framework design. Building upon this, incorporating data augmentation yields further performance gains by enriching the linguistic diversity of primitive realizations. Finally, integrating the token-level evidence identification task and representation alignment objectives during training leads to the best performance, as these components guide the primitive detection model to pinpoint evidentiary spans.

Ablation of Span Extraction To validate the necessity of *Primitive-relevant Span Extraction*, we compare our framework against a baseline without span localization. In the *w/o Extraction* setting, the model lacks access to the precise locations of semantic cues and is therefore trained solely on Global Context Instances. In contrast, our framework leverages the extracted spans to construct Background Negative Instances and Local Evidence Instances for fine-grained supervision. Using the same standard classifier architecture, the results reported in Table 5 demonstrate that explicitly isolating primitive-relevant spans through extraction is critical to achieving robust detection performance.

5.4 Further Analysis

Impact of Augmentation Scale To evaluate the impact of the augmentation scale on model performance, we vary the amount of generated transcripts and measure the resulting accuracy. The results, presented in Fig. 3, show that while even small amounts of synthetic data lead to noticeable improvements, further increasing the scale yields diminishing returns. This indicates that an excessively large synthetic dataset is not required to achieve optimal performance.

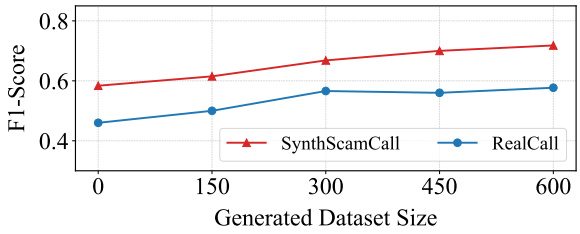

Figure 3: Influence of the Generated Dataset Size

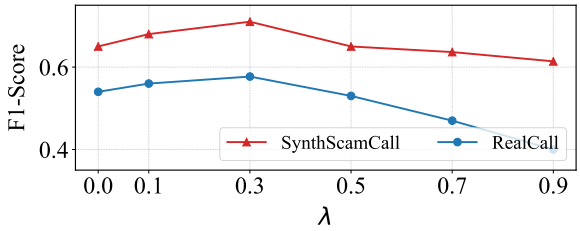

Figure 4: Influence of the alignment coefficient λ

Impact of the Alignment Weight We study the impact of the alignment weight λ by varying its value while keeping other settings fixed. As shown in Fig. 4, moderate values of λ improve performance by promoting alignment by effectively aligning global and local representations, whereas overly large values degrade performance by dominating the optimization and hindering the main classification objective. Based on these results, we set $\lambda = 0.3$ in our implementation.

6 Conclusion

In this paper, we propose a lightweight framework that anchors scam call detection on stable Semantic Primitives derived from expert knowledge. By prioritizing crucial evidentiary cues over highly variable conversational contexts, our approach effectively addresses the challenge of identifying sparse signals within noisy real-world data. Extensive experiments demonstrate that while traditional methods suffer from severe performance degradation in this challenging setting, our framework maintains superior stability and accuracy, offering a practical and efficient solution for large-scale deployment.

Limitations

Despite the efficiency of our framework, the training process relies on LLMs to generate fine-grained supervision signals. Inaccuracies or hallucinations inherent to LLMs may introduce label noise into the training corpus. Additionally, due to the difficulty of data collection, our current evaluation is limited to Chinese recruitment calls, with behaviors tightly coupled to this specific language and setting. Future work will focus on developing noise-robust learning strategies to mitigate the impact of imperfect annotations and exploring the framework's adaptability to broader domains and languages.

Ethics Consideration

We prioritize the privacy and security of users throughout this research. The RealCall dataset was collected by a cooperating online service platform solely for security auditing and fraud prevention, in accordance with the platform's User Privacy Agreement and Terms of Service accepted by users upon registration.

Before being used for academic research, the dataset was carefully de-identified. Sensitive personally identifiable information (PII), such as names, phone numbers, and detailed addresses, was masked or removed by the platform provider prior to data sharing. The data is used exclusively for research on anti-fraud methods and is not used for any improper or harmful purposes. In addition, the platform ensured that all collected transcripts do not contain high-risk content related to financial security or physical safety.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62276280, 62276279), Guangzhou Science and Technology Planning Project (No. 2024A04J9967), Guangdong Basic and Applied Basic Research Foundation (2024B1515020032), and the Key-Area Research and Development Program of Guangdong Province (2026B0101100004).

References

Mohammed Rasol Al Saidat, Suleiman Y Yerima, and Khaled Shaalan. 2024. Advancements of sms spam detection: A comprehensive survey of nlp and ml techniques. *Procedia Computer Science*, 244:248–259.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Wenbin Chen and Changqing Chen. 2024. Deep learning-based model for detecting fraudulent sms messages. In *Proceedings of the 2024 2nd International Conference on Information Education and Artificial Intelligence*, pages 346–350.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35.

Abdallah Ghourabi, Mahmood A Mahmood, and Qusay M Alzubi. 2020. A hybrid cnn-lstm model for sms spam detection in arabic and english messages. *Future Internet*, 12(9):156.

Xinxin Hu, Haotian Chen, Hongchang Chen, Shuxin Liu, Xing Li, Shibo Zhang, Yahui Wang, and Xiangyang Xue. 2023. Cost-sensitive gnn-based imbalanced learning for mobile social network fraud detection. *IEEE Transactions on Computational Social Systems*, 11(2):2675–2690.

Maximilian Ilse, Jakub Tomczak, and Max Welling. 2018. Attention-based deep multiple instance learning. In *International conference on machine learning*, pages 2127–2136. PMLR.

Ankit Kumar Jain, Kamaljeet Kaur, Naveen Kumar Gupta, and Ankit Khare. 2025. Detecting smishing messages using bert and advanced nlp techniques. *SN Computer Science*, 6(2):109.

Liming Jiang. 2024. Detecting scams using large language models. *arXiv preprint arXiv:2402.03147*.

Yufeng Kou, Chang-Tien Lu, Sirirat Sirwongwattana, and Yo-Ping Huang. 2004. Survey of fraud detection techniques. In *IEEE international conference on networking, sensing and control, 2004*, volume 2, pages 749–754. IEEE.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Jun Li, Cheng Zhang, and Lanlan Jiang. 2024. Innovative telecom fraud detection: A new dataset and an advanced model with roberta and dual loss functions. *Applied Sciences*, 14(24):11628.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, and Qing He. 2021. Pick and choose: a gnn-based imbalanced learning approach for fraud detection. In *Proceedings of the web conference 2021*, pages 3168–3177.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Haoyu Ma, Qinliang Su, Minhua Huang, and Wu Kai. 2025. Detecting continuously evolving scam calls under limited annotation: A llm-augmented expert rule framework. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5047–5068.

Oded Maron and Tomás Lozano-Pérez. 1997. A framework for multiple-instance learning. *Advances in neural information processing systems*, 10.

Dare Azeez Oyeyemi and Adebola K Ojo. 2024. Sms spam detection and classification to combat abuse in telephone networks using natural language processing. *arXiv preprint arXiv:2406.06578*.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

Zitong Shen, Kangzhong Wang, Youqian Zhang, Grace Ngai, and Eugene Yujun Fu. 2025a. Combating phone scams with llm-based detection: Where do we stand?(student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 29487–29489.

Zitong Shen, Sineng Yan, Youqian Zhang, Xiapu Luo, Grace Ngai, and Eugene Yujun Fu. 2025b. " it warned me just at the right moment": Exploring llm-based real-time detection of phone scams. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–7.

Gurjot Singh, Prabhjot Singh, and Maninder Singh. 2025. Advanced real-time fraud detection using rag-based llms. *arXiv preprint arXiv:2501.15290*.

Vincent S Tseng, Jia-Ching Ying, Che-Wei Huang, Yimin Kao, and Kuan-Ta Chen. 2015. Fraudetector: A graph-mining-based framework for fraudulent phone call detection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2157–2166.

Hong-Kui Xu, Tong-Tong Jiang, Xin Li, and 1 others. 2022. Bilstm network fraud phone recognition based on attention mechanism. *Computer Systems and Applications*, 31(3):326–332.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022a. Zerogen: Efficient zero-shot learning via dataset generation. *arXiv preprint arXiv:2202.07922*.

Jiacheng Ye, Jiahui Gao, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2022b. Progen: Progressive zero-shot dataset generation via in-context feedback. In *Findings of the association for computational linguistics: EMNLP 2022*, pages 3671–3683.

Josh Jia-Ching Ying, Ji Zhang, Che-Wei Huang, Kuan-Ta Chen, and Vincent S Tseng. 2018. Fraudetector+ an incremental graph-mining approach for efficient fraudulent phone call detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(6):1–35.

YiFan Zhang, Xue Wang, Jian Liang, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. 2023. Free lunch for domain adversarial training: Environment label smoothing. *arXiv preprint arXiv:2302.00194*.

A Details of Expert Rules

In this section, we present the detailed content of the expert judgment rules specifically derived for the online recruitment scenario. Each rule is stated in natural language and specifies a particular behavioral mode defined by the occurrence of distinctive semantic events. These behaviors serve as decisive indicators for identifying either a scam or a normal call. Notably, normal rules take precedence, meaning that a call is classified as a scam only if it explicitly triggers a fraud pattern in the absence of any normal pattern. In addition, conversations that do not trigger any of the defined rules are classified as normal by default. The complete lists of these rules, categorized by their target class, are presented in Table 10 and Table 11.

B Details of Primitives and Decision Logic

In this section, we present the definitions of the semantic primitives used in our framework in Table 12. Additionally, we illustrate the corresponding logical decision functions derived from the expert rules in Table 13.

C Additional Framework Details

C.1 Implementation Details of Primitive and Decision Logic Acquisition

As outlined in Section 4.2, the acquisition phase yields a set of semantic primitives $\mathcal{P}$ and the corresponding decision logic $Logic(\cdot)$. In our practical implementation, to support effective downstream data curation—specifically Primitive-Relevant Span Extraction and Data Augmentation—we further refine the representation of each primitive. Since relying solely on concise primitive names may introduce semantic ambiguity and limit the LLM's ability to accurately locate evidence or generate valid variations, we additionally formulate a detailed semantic definition for each primitive $p \in \mathcal{P}$.

We also provide an automated solution by leveraging LLMs to parse and formalize domain-specific rules into the required structured format. The relevant prompts utilized for primitive extraction and logic parsing are presented in Table 14.

C.2 Implementation Details of Primitive-Relevant Span Extraction

As outlined in 4.3.1, we employ an LLM as a zero-shot extractor to locate evidence spans within the call transcripts. Guided by the precise definitions derived in the previous step, the model is instructed to identify minimal text spans that remain semantically complete for each primitive. The specific prompts are presented in Table 15.

C.3 Implementation Details of Data Augmentation

As outlined in 4.3.2, our data augmentation strategy involves two key steps: Primitive Paraphrasing and Semantic Injection.

First, to enrich the linguistic diversity of the target primitives, we instruct the LLM to generate multiple conversational variations based on the semantic description of each primitive. Subsequently, we employ the LLM to seamlessly inject these generated variations into the background transcripts, ensuring that the insertion maintains context coherence. The specific prompts used for these tasks are presented in Table 16.

D Experiment details

D.1 Details of Datasets

The **RealCall** dataset is collected from real-world voice interactions on a large online recruitment platform. The call transcripts were retained by the platform as part of routine security monitoring to detect and prevent malicious activities, in accordance with internal operational policies. Before being shared for academic research, the raw transcripts were processed to remove sensitive information, with private details filtered out or replaced by generic placeholders. For dataset construction, the training and testing sets were drawn from different time periods. This temporal split reflects practical deployment settings, where conversation contexts change over time, and allows evaluation of how well models perform on data collected after the training period.

For the **SynthScamCall** benchmark, we followed the dataset configuration described in Ma et al. (2025).

D.2 Training Details

All lightweight models were implemented using PyTorch and the HuggingFace Transformers library with the AdamW optimizer (Loshchilov and Hutter, 2017). For all components involving Large Language Models, we accessed Qwen3-max (Yang et al., 2025) and Deepseek-v3 (Liu et al., 2024) via the Bailian Model Studio API. We set the sampling

temperature to 0.7 for all LLM inference tasks. For all models requiring training, we conducted experiments on a single NVIDIA RTX 3090 GPU, reporting the average results across three independent runs.

Training Configurations. Regarding the baselines, we employed Focal Loss (Rasooli and Tetreault, 2015) to mitigate class imbalance, with hyperparameters set to $\alpha = 2$ and $\beta = 0.6$. Specifically, the standard Finetuned-BERT was trained using a batch size of 64 and a learning rate of 2e-5 for 3 epochs. For MIL, we employed an attention-based implementation, using a batch size of 32 and a learning rate of 2e-5, training for 4 epochs. For ELS, given the complexity of domain adversarial training, we extended the training duration to 15 epochs, with a batch size of 32 and a learning rate of 1e-5. Finally, for RNP, we re-implemented the method using BERT as the backbone; both the generator and encoder were trained with a learning rate of 1e-5 and a batch size of 32 for 4 epochs, while the sparsity and continuity regularization coefficients were set to 1e-5 and 2e-5, respectively.

In terms of the generation-based methods, For ZeroGen, we generated 2,000 additional samples using prompts containing rule descriptions and one real example. For ProGen, we held out 500 samples from the training set as a validation set, and then used prompts with rule descriptions, one real example, and one generated example to generate 2,000 additional samples. For ELS, we used prompts that explicitly instructed the model to produce samples across different scenarios to generate 2,000 cross-domain samples. We then used these samples to train DANN (Ganin et al., 2016) equipped with ELS for domain-adversarial learning.

Prompt Designs for Baselines. In addition to our proposed framework, several baselines also leverage LLMs for data generation. To ensure a fair comparison and high-quality synthesis, we designed specific prompts for these methods. The exact prompts utilized for these baselines are also listed in Table 17.

E Additional Experiment Result

E.1 Visualization of Distribution Shift

To intuitively illustrate the distribution shift between the training and testing sets, we visualize the representations of the call transcripts. Specifically, we extract the embeddings using the bert-base-chinese model and apply t-SNE for dimensionality reduction. As shown in Figure 5, there is a clear distribution shift between the training and testing samples.

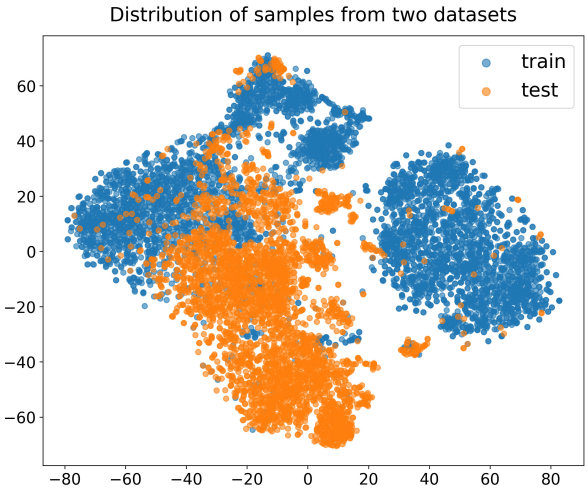

Figure 5: t-SNE visualization of the transcript embeddings.

E.2 Impact of the Number of Semantic Variations

Number of Variations	RealCall	SynthScamCall
0	0.444	0.583
25	0.561	0.667
50 (Paper Default)	0.578	0.718
75	0.582	0.715
100	0.579	0.722

Table 6: Performance comparison with different numbers of variations (K) for data augmentation.

To evaluate the impact of the variation quantity on model performance, we conducted experiments with different numbers of semantic paraphrases K. The results are presented in Table 6. As K increases from 0 to 50, the performance improves substantially on both datasets. This improvement is attributed to the efficacy of data augmentation, which exposes the lightweight model to a diverse range of expressions beyond the original training data, thereby enhancing its generalization capability regarding the core semantics of the primitives.

However, further increasing the number of variations to 75 or 100 results in only marginal performance gains. This plateau arises because the core definitions of the Semantic Primitives in our framework are inherently precise and singular. Consequently, generating an excessive number of variations (e.g., exceeding 50) for a single atomic primi-

tive tends to produce expressions that are either rare in daily speech or increasingly ambiguous, offering limited additional value to the training process. Therefore, we selected $K = 50$ as the optimal setting to balance computational efficiency with detection performance.

E.3 Impact of the Number of Augmented Primitives

Augmentation Setting	RealCall	SynthScamCall
6 Primitives (Paper Default)	0.578	0.718
12 Primitives (All)	0.583	0.721

Table 7: Performance comparison with different numbers of primitives for data augmentation.

Type	Description
1	Mentions a large number of recruitment positions.
2	Mentions security guard positions.
3	Mentions that room and board are provided.
4	Mentions onboarding training.
5	Mentions arranging work nearby.
6	Mentions having branches nationwide.

Table 8: Descriptions of the primitives used for augmentation.

To evaluate the impact of the number of augmented primitives on model performance, we conducted experiments comparing our default augmentation strategy against augmenting all available primitives. As shown in Table 7, we evaluated these configurations across different datasets.

In our default setting, we strategically selected six specific primitives for augmentation, as detailed in Table 8. This decision is primarily based on the observation that the occurrence frequency of different semantic primitives in our training dataset varies significantly, forming a long-tail distribution. Primitives associated with standard recruitment procedures (e.g., explicitly specifying job titles or inviting for an interview) appear frequently within the large volume of normal calls. This high frequency allows the model to learn these features effectively without requiring additional augmentation. Conversely, other primitives appear sparsely due to the inherent data distribution. This sparse category includes specific job types with lower sampling frequency and, more notably, core scam indicators that are naturally rare due to the scarcity of scam samples. To address this imbalance while maintaining computational efficiency, we explicitly targeted the six least frequent primitives for augmentation.

To empirically validate this design choice, we also experimented with extending the augmentation process to include all 12 primitives. The experimental results indicate that augmenting all primitives yields only marginal performance gains. This demonstrates that the model has already learned sufficient features for the high-frequency primitives directly from the original data. Therefore, further augmentation is unnecessary.

E.4 Impact of Multiple Instance Learning Window Size

Window Size	RealCall	SynthScamCall
10	0.091	0.234
25	0.116	0.253
50 (Paper Default)	0.141	0.246
75	0.131	0.261
100	0.143	0.242

Table 9: Performance comparison across different window sizes.

We evaluated the window size of the Multiple Instance Learning baseline. As shown in Table 9, the model performance remains consistently low across all settings and exhibits only marginal fluctuations. This result demonstrates that the underperformance of MIL does not stem from improper hyper-parameter selection. Standard MIL formulations typically assume that a label is determined solely by the presence or absence of isolated suspicious segments. Such an assumption is insufficient for accurate judgment because the final verdict in complex fraud scenarios depends on the logical combination of multiple segments. Furthermore, the diverse and lengthy contexts in call transcripts often lead the MIL classifier to identify spurious correlations that do not indicate actual fraud. Regardless of window size adjustments, MIL fails to function effectively and maintains a significant performance gap compared to our proposed framework.

Scam Rule Content
(1) 通话文本没有说明具体岗位名称，且提到招聘岗位多，则为诈骗通话。补充说明：“招聘岗位多”是指文中出现“工作比较多”、“岗位就是比较多”、“岗位挺多的”等类似表达的情况，注意是明确提到岗位很多而不是几个岗位。 **Trans:** If the call transcript does not specify a job title and mentions many recruitment positions, it is classified as a fraudulent call. Supplementary note: "Many recruitment positions" refers to cases where expressions like "many jobs," "positions are just many," or "quite a lot of positions" appear in the text. Note that it specifically refers to a large number of positions, not just a few.
(2) 通话文本中没有提及岗位是保安、配送类职业、司机，但提到可以包吃包住为诈骗通话。补充说明：“包吃包住”是指文中出现“我们这边包吃包住的”、“我们这边提供吃住”等类似表达的情况。 **Trans:** If the call transcript does not mention that the position is for security, delivery, or driving, but mentions that room and board are included, it is classified as a fraudulent call. Supplementary note: "Room and board included" refers to cases where expressions like "we include room and board here" or "we provide food and living" appear in the text.
(3) 通话文本中没有提及岗位是配送类职业，但提到需要入职培训为诈骗通话。补充说明：“入职培训”是指文中出现“我们会有岗前培训的”、“这边都是有人带的，然后会有培训”等类似表达的情况。 **Trans:** If the call transcript does not mention the position is for delivery services, but mentions mandatory onboarding training, it is classified as a fraudulent call. Supplementary note: "Onboarding training" refers to cases where expressions like "we will have pre-job training" or "someone will lead you here, then there will be training" appear in the text.
(4) 通话文本中没有提及岗位是司机、配送类职业，但提及可以就近安排工作为诈骗通话。补充说明：“就近安排”是指文中出现“给你就近安排”等类似表达的情况，注意必须包含“就近”关键语义，不能是模糊的“安排”或“分配”。 **Trans:** If the call transcript does not mention the position is for a driver or delivery services, but mentions that work can be arranged nearby, it is classified as a fraudulent call. Supplementary note: "Nearby arrangement" refers to cases where expressions like "arrange nearby for you" appear in the text. Note that it must contain the key semantic of "nearby" and cannot be vague terms like "arrangement" or "assignment."
(5) 通话文本中没有提及岗位是司机、配送类职业，但提及全国都有分公司为诈骗通话。补充说明：“全国都有分公司”是指文中出现“我们是xx公司的，全国设有多个分公司”、“我们店在全国各地都有分店”、“在全国有xx家分公司”等类似表达的情况，注意必须强调“全国范围”，而不是仅提到某些具体城市或地区。 **Trans:** If the call transcript does not mention the position is for a driver or delivery services, but mentions having branches nationwide, it is classified as a fraudulent call. Supplementary note: "Branches nationwide" refers to cases where expressions like "we are company xx, with multiple branches nationwide," "our shops have branches all over the country," or "there are xx branch companies nationwide" appear in the text. Note that it must emphasize "nationwide scope," not just specific cities or regions.

Table 10: Detailed Content of Expert Rules for Scam Calls

Normal Rule Content
(1) 文本中提到招聘方的详细的地址信息是正常通话。补充说明：文中出现“我们在xx城xx区”、“位于xx大厦xx楼”、“地点是xx广场附近”等类似表达说明是正常通话，注意公司名称和店名以及省市都不算地名。 **Trans:** If the text mentions detailed address information of the recruiter, it is a legitimate call. Supplementary note: Expressions like “we are in xx district, xx city,” “located on floor xx, xx building,” or “location is near xx square” indicate a legitimate call. Note that company names, shop names, and mere province/city names do not count as detailed addresses.
(2) 招聘方邀请面试为正常通话。补充说明：文中出现“什么时候方便过来面试”、“过来面谈一下”等类似表达说明是正常通话，需明确提及面试而不是电话或线上联系。 **Trans:** If the recruiter invites the candidate for an interview, it is a legitimate call. Supplementary note: Expressions like “when is it convenient to come for an interview” or “come over for a talk” indicate a legitimate call. It must explicitly mention an interview rather than phone or online contact.
(3) 招聘方邀请线下了解为正常通话。补充说明：文中出现“过来看看”等类似表达说明是正常通话，需明确提及线下参观了解而不是电话或线上联系。 **Trans:** If the recruiter invites the candidate to visit in person to understand the job, it is a legitimate call. Supplementary note: Expressions like “come over and take a look” indicate a legitimate call. It must explicitly mention an offline visit rather than phone or online contact.

Table 11: Detailed Content of Expert Rules for Normal Calls

Semantic Primitive	Content
p_1	提及入职培训。**Trans:** Mentions onboarding training.
p_2	提及就近安排工作。**Trans:** Mentions arranging work nearby.
p_3	提及包吃包住。**Trans:** Mentions that room and board are provided.
p_4	提及全国范围设有分公司。**Trans:** Mentions having branches nationwide.
p_5	提到招聘岗位多。**Trans:** Mentions a large number of recruitment positions.
p_6	通话文本明确具体岗位名称。**Trans:** The call explicitly specifies job titles.
p_7	提及司机岗位。**Trans:** Mentions driver positions.
p_8	提及配送类职业。**Trans:** Mentions delivery-related occupations.
p_9	提及保安岗位。**Trans:** Mentions security guard positions.
p_{10}	招聘方邀请线下参观了解。**Trans:** The recruiter invites for an offline visit.
p_{11}	招聘方邀请面试。**Trans:** The recruiter invites for an interview.
p_{12}	招聘方提及的具体地址信息。**Trans:** The recruiter mentions specific address information.

Table 12: The Set of Semantic Primitives

Behavior Pattern	Logic Function
Scam Rule 1	$\neg z_6 \wedge z_5$
Scam Rule 2	$\neg z_7 \wedge \neg z_8 \wedge \neg z_9 \wedge z_3$
Scam Rule 3	$\neg z_8 \wedge z_1$
Scam Rule 4	$\neg z_7 \wedge \neg z_8 \wedge z_2$
Scam Rule 5	$\neg z_7 \wedge \neg z_8 \wedge z_4$
Normal Rule 1	z_{10}
Normal Rule 2	z_{11}
Normal Rule 3	z_{12}

Table 13: Logical Expressions for Detection

Prompt Name	Prompt Content
Rule Decomposition	请根据以下要求，将给定的规则转换为标准的逻辑表达式，并以指定的JSON 格式输出。任务说明：1. 分析以下已概括的规则：{rule}。2. 将该规则分解为若干个原子条件，并使用逻辑运算符（AND、OR、NOT）组合成一个完整的逻辑表达式。每个条件必须用方括号括起来，例如：[条件内容]。否定条件应表示为：NOT[条件内容]。3. 输出必须是一个JSON 对象，且仅包含一个键："output"。格式示例：{{"output": "(NOT [通话文本明确具体岗位名称]) AND [提到招聘岗位多]"}}。不要添加任何额外字段或解释性文字。4. 注意事项：确保逻辑表达式完整覆盖原规则的判断逻辑；条件描述应简洁明确，避免冗余或模糊用语；不要包含原始规则中的示例或非判断性说明。请开始输出。
	Trans: Please convert the given rule into a standard logical expression according to the requirements below and output it in the specified JSON format. **Task Instructions:** 1. Analyze the following summarized rule: {rule}. 2. Decompose the rule into atomic conditions and combine them using logical operators (AND, OR, NOT) to form a complete logical expression. Each condition must be enclosed in square brackets, e.g., [Condition Content]. Negated conditions should be denoted as: NOT[Condition Content]. 3. The output must be a single JSON object containing exactly one key: "output". Example format: {{"output": "(NOT [Call transcript specifies exact job title]) AND [Mentions multiple positions]"}}. Do not add any extra fields or explanatory text. 4. Notes: Ensure the logical expression fully covers the decision logic of the original rule. Condition descriptions should be concise and unambiguous. Do not include examples or non-judgmental text found in the original rule. Please begin.
Primitive Definition	规则：{rule}。要求：这条规则涉及的要点包括{primitive 1}.....{primitive K}，严格根据规则的内容，对每个要点进行定义和描述，输出格式为：[{"Primitive":要点名称, "Definition":要点定义}]。结果只输出json格式。
	Trans: Rule: {rule}. Requirements: The key points involved in this rule include {primitive 1}...{primitive K}. Strictly based on the content of the rule, define and describe each point. The output format is: [{"Primitive": Point Name, "Definition": Point Definition}]. Output the result in JSON format only.

Table 14: List of Prompts for Primitive and Decision Logic Acquisition

Prompt Name	Prompt Content
Span Extraction	输入：{call transcript}。要求：你是一个信息提取专家。请根据{primitive}和{definition of primitive}，执行以下任务：1. 理解命题：结合定义准确理解该命题的语义要求。2. 提取文本片段：在文本中找出所有语义上严格满足该命题的最短完整句子或子句片段。要求：片段必须语义完整，能独立表达一个意思；尽可能简短，避免冗余上下文；如果该命题在文本中无匹配内容，则span 字段应为空列表[]。3. 输出格式：返回一个JSON 对象，包含primitive 和span 两个字段。
	Trans: Input: {call transcript}. Requirements: You are an information extraction expert. Please execute the following tasks based on {primitive} and {definition of primitive}: 1. Understand Primitive: Accurately grasp the semantic requirements based on the definition. 2. Extract Text Spans: Identify all shortest complete sentences or clauses in the text that strictly satisfy the primitive semantically. Requirements: Spans must be semantically complete and independent; be as concise as possible to avoid redundant context; if there is no matching content, the 'span' field should be an empty list []. 3. Output Format: Return a JSON object containing 'primitive' and 'span' fields.

Table 15: Prompt for Primitive-Relevant Span Extraction

Prompt Name	Prompt Content
Primitive Paraphrasing	这是一则招聘通话的关键语义及其说明：{primitive}{definition of primitive}。请生成{num_generate} 种通话中的不同的表达。 **Trans:** Here is a key semantic point of a recruitment call and its description: {primitive} {definition of primitive}. Please generate {num_generate} different conversational expressions for this point.
Semantic Injection	输入：{call transcript}。要求：在不改动原文任何原有段落的前提下，将片段{span}自然地融入到该通话内容中。确保原段落结构、语序和用词完全保持不变，仅在合适的位置插入该短语，使其语义连贯、符合上下文逻辑。 **Trans:** Input: {call transcript}. Requirement: Without altering any original paragraphs of the text, naturally integrate the fragment {span} into the call content. Ensure the original structure, word order, and wording remain completely unchanged, inserting the phrase only in an appropriate position to maintain semantic coherence and contextual logic.

Table 16: List of Prompts for Data Augmentation Strategy in Our Framework

Prompt Name	Prompt Content
ZeroGen	以下是用于判断通话文本是否为正常通话或异常通话的规则：{normal rule 1}.....{normal rule K}，{scam rule 1}.....{scam rule K} 下面是一则示例通话文本（其类别为：{type of call transcript}）： {original call transcript} 请根据以上规则与示例文本的风格、主题、语气、内容结构，生成一个全新的、与示例同类别的通话样本。 **Trans:** Here are the rules for judging whether a call transcript is normal or scam: {normal rule 1}...{normal rule K}, {scam rule 1}... {scam rule K}. Below is a sample call transcript (Category: {type of call transcript}): {original call transcript} Please generate a brand new call sample of the same category as the example, based on the above rules and the style, theme, tone, and content structure of the sample text.
ProGen	以下是用于判断通话文本是否为正常通话或异常通话的规则：{normal rule 1}.....{normal rule K}，{scam rule 1}.....{scam rule K} 下面是两则示例通话文本（其类别为：{type of call transcript}）： {original call transcript}{generated call transcript} 请根据以上规则与示例文本的风格、主题、语气、内容结构，生成一个全新的、与示例同类别的通话样本。 **Trans:** Here are the rules for judging whether a call transcript is normal or scam: {normal rule 1}...{normal rule K}, {scam rule 1}... {scam rule K}. Below are two sample call transcripts (Category: {type of call transcript}): {original call transcript}{generated call transcript} Please generate a brand new call sample of the same category as the example, based on the above rules and the style, theme, tone, and content structure of the sample text.
ZeroGen for ELS	以下是用于判断通话文本是否为正常通话或异常通话的规则：{normal rule 1}.....{normal rule K}，{scam rule 1}.....{scam rule K} 下面是一则示例通话文本（其类别为：{type of call transcript}）： {original call transcript} 请根据以上规则与示例文本的风格、语气，生成一个不同场景的、与示例同类别的通话样本。 **Trans:** Here are the rules for judging whether a call transcript is normal or scam: {normal rule 1}...{normal rule K}, {scam rule 1}... {scam rule K}. Below is a sample call transcript (Category: {type of call transcript}): {original call transcript} Please generate a call sample of the same category as the example but in a different scenario, based on the above rules and the style and tone of the sample text.

Table 17: List of Prompts for Data Generation Strategies Used by Baselines